An Adaptive Finite Volume Method for the Incompressible Navier-Stokes Equations in Complex Geometries

David Trebotich * and Daniel T. Graves

Computational Research Division, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720, USA

Abstract

We present an adaptive, finite volume algorithm to solve the incompressible Navier-Stokes equations in complex geometries. The algorithm is based on the embedded boundary method in which finite volume approximations are used to discretize the solution in cut cells that result from intersecting the irregular boundary with a structured Cartesian grid. This approach is conservative and reduces to a standard finite difference method in grid cells away from the boundary. We solve the incompressible flow equations using a predictor-corrector formulation. Hyperbolic advection terms are obtained by higher-order upwinding without the use of extrapolated data in covered cells. The small cell stability problem associated with explicit embedded boundary methods for hyperbolic systems is avoided by the use of a volume-weighted scheme in the advection step, and is consistent with construction of the right-hand side of the elliptic solvers. The Helmholtz equations resulting from viscous source terms are advanced in time by the Crank-Nicolson method which reduces solver runtime compared to other second-order time integrators by a half. Incompressibility is enforced by a second-order approximate projection method that makes use of a new conservative cell-centered gradient in cut cells which is consistent with the volume-weighted scheme. The algorithm is also capable of block structured adaptive mesh refinement to increase spatial resolution dynamically in regions of interest. The resulting overall method is second-order accurate for sufficiently smooth problems. In addition, the algorithm is implemented in a high performance computing framework and can perform structured-grid fluid dynamics calculations at unprecedented scale and resolution, up to 262,144 processor cores. We demonstrate robustness and performance of the algorithm by simulating incompressible flow for a wide range of Reynolds numbers in two and three dimensions: Stokes and low Reynolds number flows in both constructed and image data geometries ($Re \ll 1$ to $Re = 1$), flow past a cylinder ($Re = 300$), flow past a sphere ($Re = 600$) and turbulent flow in a contraction ($Re = 6300$).

Key words: incompressible Navier-Stokes, embedded boundary method, finite volume method, cut cell method, projection method, adaptive mesh refinement

* Corresponding author.

Email addresses: dptrebotich@lbl.gov (David Trebotich), dtgraves@lbl.gov (Daniel T. Graves).

1 Introduction

In this paper, we describe a conservative, high resolution algorithm for the incompressible Navier-Stokes equations in complex geometries. The primary outcome of this work is a simulation capability that can be applied to a wide range of flows where high resolution is sought—from low Reynolds number flow in geologic or engineered porous media, for example, to direct numerical simulation of turbulence. Our approach is based on an adaptive, finite volume embedded boundary method. In the context of a complete description of the overall algorithm, we present several novel numerical techniques including: a volume-weighted scheme for finite volume discretizations that avoids the small cell problem associated with hyperbolic solvers based on cut cell methods; and a stable second-order time integration method that is faster than other second-order schemes used in the context of embedded boundary methods. We demonstrate second-order convergence of the algorithm. We apply the algorithm to benchmark flow past a cylinder in 2D and 3D, flow past a sphere in 3D, high Reynolds number flow in a 2D contraction, as well as Stokes flow and low Reynolds number flow in packed bed geometries and realistic subsurface materials. We also demonstrate the adaptive mesh refinement capability of the algorithm as well as scalable performance to 262,144 processor cores.

1.1 Equations of Motion

We consider the incompressible Navier-Stokes equations with constant density:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = -\nabla p + \nu \Delta \mathbf{u} \tag{1.1}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{1.2}$$

where $\mathbf{u}$ is the fluid velocity, ∇p is the pressure gradient and ν is the kinematic viscosity. To close the system, we specify boundary conditions for a bounded inflow-outflow problem. For example, for flow in the x-direction in a two-dimensional channel, the boundary conditions are, at inflow,

$$\mathbf{u} = \left(\frac{3}{2}\bar{u}(1 - y^2/a^2), 0\right), \quad \frac{\partial p}{\partial x} = 0, \tag{1.3}$$

where $\bar{u}$ is average inflow velocity and a is half the width of the channel in y-direction; at solid walls,

$$\mathbf{u} = 0, \quad \frac{\partial p}{\partial y} = 0; \tag{1.4}$$

and at outflow,

$$\frac{\partial \mathbf{u}}{\partial x} = 0, \quad p = 0. \tag{1.5}$$

Given initial conditions $\mathbf{u}^0 = \mathbf{u}(\mathbf{x}, 0)$, $p^0 = p(\mathbf{x}, 0)$, the system of equations defined by (1.1)-(1.5) constitutes an initial boundary value problem (IBVP) that can be solved by a variety of methods (e.g., immersed boundary [28], ghost fluid [18], discontinuous Galerkin [22]). We are ultimately interested in efficient, scalable computations to obtain high resolution for a wide range of Reynolds number flows in complex geometries. We address this problem with a predictor-corrector projection formulation based on a finite volume, embedded boundary method with adaptive mesh refinement.

1.2 Numerical Approach

1.2.1 Embedded boundary method

Cartesian grid methods have become an increasingly popular modeling approach to solving partial differential equations (PDEs) in complex geometries. There are several Cartesian grid approaches (e.g., immersed boundary [44], immersed interface [30], ghost fluid [18]), however, we focus on the cut cell approach which is based on finite volume approximations. A cut cell, or embedded boundary, method refers to a finite volume discretization in irregular cells on a Cartesian grid that result from the intersection of the boundary and the rectangular cells of the grid. Conservative numerical approximations to the solution can be found from discrete integration over the non-rectangular control volumes, or cut cells, with fluxes located at centroids of the edges or faces of a control volume. This approach has been used as the basis for second-order accurate methods for elliptic, parabolic and hyperbolic PDEs in two and three dimensions [21,34,15].

One of the advantages of the method is that the problem of generating the description of complex geometry on the grid (starting from, for example, surface tessellations produced by a CAD system, or implicit function representation of x-ray microtomography images) has been made more tractable [1,31]. Another advantage of the embedded boundary method is that it is amenable to adaptive mesh refinement (AMR) [8]. Block structured AMR is a technique to add grid resolution efficiently and dynamically in areas of interest while leaving the rest of the domain at a coarser resolution. AMR was originally applied to finite difference methods for inviscid shock hydrodynamics [9], and has been extended to inviscid, incompressible flow [32] and viscous flow [3,33] in rectangular domains. AMR has been combined with embedded boundary methods to model inviscid and viscous compressible flows in complex geometries [43,15,19].

For incompressible flows, embedded boundary methods have been mostly applied to inviscid flows (e.g., [2]). As attractive as cut cell methods are for efficient gridding of complex geometries, these methods are still gaining ground in the engineering community for modeling of incompressible viscous flows, perhaps due to the high resolution that is required in and around cut cells to resolve viscous boundary layers. Therefore, AMR and high performance computing have become necessary partners

for cut cell methods to be effective 3D modeling tools. Furthermore, discussion of such methods usually centers around the "small cell problem" due to the arbitrary nature of the cut cell approach; also of importance are accuracy of gradients and higher-order strategies.

Several methods have been proposed for incompressible viscous flow using the cut cell approach. In [60], a single-grid (non-adaptive) finite volume method is used for 2D incompressible viscous flows and is demonstrated on an array of cylinders in a channel. In [45] adaptivity is combined with a volume-of-fluid method and applied to practical engineering problems in 3D. Cell-merging is used to treat the small cell problem. Second-order accuracy is demonstrated with particular attention given to the pressure gradient. In [27] a cut cell method on a staggered grid is applied to a moderate Reynolds number flow in 3D. A "cell-linking" method is proposed that links small cells with a master cell, placing the cell a small distance from the master and inducing a high diffusion flux which forces the two velocities to take the same value. A cell-merging technique was used in [13] as part of a cut cell projection method. In a precursor to the work presented here, an embedded boundary method was used in [53] to model fluid-particle flow through a packed bed geometry. This work was later generalized to AMR in a computationally efficient framework using novel stenciling techniques in cut cells [54]. The small cell problem was addressed by a linear hybridization of conservative and non-conservative estimates of the convective derivative akin to [11,6] with redistribution of the unconserved mass.

1.2.2 Projection method

Projection methods address the time-discretization issue of the constrained evolution equations of incompressible flow. These methods are based on the Hodge-Helmholtz decomposition of a vector field into a divergence-free part and a gradient of a scalar field, effectively separating the vortical dynamics induced by a viscous, divergence-free velocity field from the potential flow problem. Projection methods have taken several different paths since Chorin's original method was introduced [12], primarily depending on the choice of scheme for higher-order discretization of the nonlinear advection term (e.g., [26,23]). Our approach is based on the work of Bell, Colella and Glaz (BCG) [4], and the family of methods that followed (e.g., [29,3,50,33]). The BCG method makes use of high resolution finite difference methods for hyperbolic PDEs, such as Godunov or upwinding schemes, combined in a fractional step approach with fast iterative methods for elliptic and parabolic PDEs to achieve second-order spatial and temporal accuracy. BCG was made more robust for larger CFL numbers with the introduction of an intermediate "MAC" (marker-and-cell) projection in the advection step [5]. In [50] the BCG method was extended to time-dependent domains using quadrilateral, mapped grids and a consistent decomposition of the velocity field which standardized the implementation of boundary conditions for projection methods. The projection method was generalized to a non-adaptive embedded boundary approach for time-dependent domains in [36].

In this paper we combine these methods—adaptive, finite volume and projection—using the predictor-corrector projection formulation in [50], the adaptive approach

in [33] and the computational fluid dynamics tools for adaptive embedded boundary methods in [54] to solve the incompressible Navier-Stokes equations in complex geometries. The algorithm is implemented in the Chombo software framework which supports adaptive, embedded boundary methods and also enables large scale computations (`chombo.lbl.gov`). The resulting algorithm is conservative, second-order accurate and scalable to 262,144 processor cores. The central new idea of the algorithm is a volume-weighted scheme that avoids the small cell problem associated with explicit embedded boundary methods and leads to better stability properties than previous approaches in [54,15,53].

We organize the discussion of the algorithm as follows. The finite difference algorithm is described in its entirety in §2. The embedded boundary, finite volume method is described in §3 for the case of cut cells where the discretization requires special stencils that differ from the finite difference method. For ease of exposition, the algorithm is described in 2D; the 3D discretization is included if it is not an obvious extension from 2D. We include brief descriptions of algorithm modifications needed for AMR throughout the discussion, and particularly for hyperbolic and elliptic discretizations near coarse-fine interfaces. Accuracy of the method, performance measurements and simulation results are presented in §4. Conclusions are summarized and discussed in §5.

2 Algorithm Discretization

A 2^{nd}-order accurate in time discretization of the evolution equation (1.1) is as follows:

$$U^{n+1} = U^n + \Delta t(\nu \Delta U^{n+1/2} - (U \cdot \nabla)U^{n+1/2} - \nabla p^{n+1/2})$$

where U^n is an approximation of the velocity field at the discrete time $t^n = t^{n-1} + \Delta t$. We choose a 2^{nd}-order upwind method for hyperbolic terms, a 2^{nd}-order implicit discretization of parabolic terms, and an approximate projection method to enforce incompressibility with special centering of the pressure gradient. We combine these methods in a semi-implicit predictor-corrector formulation based on [50] to advance the solution.

2.1 Temporal discretization

The momentum equation (1.1) can be formulated as a parabolic equation of the form $U_t = \mathcal{L}(U) + f(U)$, where $\mathcal{L}$ is a 2^{nd}-order elliptic operator such as Laplacian. Second-order accuracy in time can be achieved by the Crank-Nicolson method for parabolic equations as in [4]. It has been previously reported that, in the presence of embedded boundaries, the Crank-Nicolson scheme is unstable for parabolic equations, and in particular, when the embedded boundary is moving, coefficients are strongly varying or discontinuities exist in the solution [34]. Instead, the Runge-Kutta method of [55] is

recommended to achieve second-order accuracy. In practice, we have not experienced such instabilities with Crank-Nicolson for stationary boundaries, nor in current work with moving boundaries (e.g., [36]). Furthermore, significant computational savings are gained from the use of Crank-Nicolson which requires only D (number space dimensions) solutions to the Helmholtz problem while the 2^{nd}-order Runge-Kutta method as described in [55] requires $2D$ solutions to the Helmholtz problem.

The Crank-Nicolson discretization is as follows:

$$(I - \frac{\nu\Delta t}{2}\Delta)U^{n+1,*} = (I + \frac{\nu\Delta t}{2}\Delta)U^n + \Delta t f^{n+1/2} \tag{2.6}$$

$$f^{n+1/2} = -(U \cdot \nabla)U^{n+1/2} - \nabla p^{n-1/2}. \tag{2.7}$$

The intermediate velocity, $U^{n+1,*}$, in (2.6) is a 2^{nd}-order approximation to the solution that satisfies the boundary conditions but does not necessarily satisfy the incompressibility constraint due to the lagged pressure gradient in (2.7).

2.2 Projection formulation

The projection method [12] is used to enforce incompressibility in discretization (2.6). In general, a smooth vector field, $\mathbf{w}$, on a simply connected domain, Ω, can be orthogonally decomposed into a divergence-free component $\mathbf{w}_d$ and a gradient of a scalar potential, ψ

$$\begin{aligned} \mathbf{w} &= \mathbf{w}_d + \nabla\psi \\ \nabla \cdot \mathbf{w}_d &= 0 \\ \Delta\psi &= \nabla \cdot \mathbf{w} \end{aligned}$$

with boundary conditions $\mathbf{w}_d \cdot \mathbf{n} = 0$ and $\frac{\partial\psi}{\partial n} = \mathbf{w} \cdot \mathbf{n}$ on $\partial\Omega$. We can apply a discrete version of the projection to the discretization (2.6) to obtain a divergence-free velocity and pressure gradient:

$$\begin{aligned} U^{n+1} &= \mathbf{P}(W) \\ \nabla p^{n+1/2} &= \frac{1}{\Delta t}\mathbf{Q}(W) \\ W &= U^{n+1,*} + \Delta t \nabla p^{n-1/2} \end{aligned}$$

where $\mathbf{Q} = \mathbf{G}\mathbf{L}^{-1}\mathbf{D}$, $\mathbf{P} = \mathbf{I} - \mathbf{Q}$, and $\mathbf{L}$, $\mathbf{D}$ and $\mathbf{G}$ are discrete representations of the Laplacian, divergence and gradient, respectively. These projection operations procedurally reduce to solution of the Poisson problem and an update of the velocity and pressure gradient by the gradient of the solution to the Poisson problem

$$\mathbf{L}\phi = \mathbf{D}(W) \tag{2.8}$$
$$U^{n+1} = W - \mathbf{G}(\phi) \tag{2.9}$$
$$\nabla p^{n+1/2} = \frac{1}{\Delta t}\mathbf{G}(\phi). \tag{2.10}$$

We note that the projection target, W, contains the intermediate velocity augmented by the lagged pressure gradient resulting in a pressure formulation with improved stability in comparison to the pressure correction formulation in [50].

The form of $\mathbf{G}$, and thus, $\mathbf{L}$, depends on the centering of the projection target, W, which is cell-centered in this case. However, the discretization of divergence, based on the discrete form of the divergence theorem, is applied as a sum of differences of *face-centered* data in each direction. In compact notation, we have

$$\mathbf{D}(W)_{\boldsymbol{i}} = \frac{1}{h}(\sum_{d=1}^{D}(W_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d} - W_{\boldsymbol{i}-\frac{1}{2}\hat{e}^d}). \tag{2.11}$$

If we also consider that the gradient is applied to the cell-centered solution to Poisson's equation, $\phi_{\boldsymbol{i}}$, then $\mathbf{D}$ and $\mathbf{G}$ are not discrete adjoints, and $\mathbf{L} \neq \mathbf{DG}$. This projection is, therefore, approximate and $\mathbf{D}(\mathbf{P}(W)) = O(h^2)$, the same magnitude as the truncation error. Also, the operator is not idempotent, i.e., $\mathbf{P}^2 \neq \mathbf{P}$.

If the divergence and gradient are discrete adjoints and $\mathbf{L} \equiv \mathbf{DG}$, then the projection is discretely exact, i.e., $\mathbf{D}(\mathbf{P}(W)) = 0$ [12]. This is the case of the so-called MAC projection, defined to be $\mathbf{P}^{mac} \equiv (\mathbf{I} - \mathbf{Q}^{mac})$, and $\mathbf{Q}^{mac} \equiv \mathbf{G}^{mac}(\mathbf{L}^{mac})^{-1}\mathbf{D}$. The discrete Laplacian operator can then be defined as the conservative divergence of the face-centered gradient:

$$\mathbf{L}_{\boldsymbol{i}} \equiv \mathbf{D}(\mathbf{G}^{mac}(\phi))_{\boldsymbol{i}} = \frac{1}{h}(\sum_{d=1}^{D}(\mathbf{G}^{mac,d}(\phi)_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d} - \mathbf{G}^{mac,d}(\phi)_{\boldsymbol{i}-\frac{1}{2}\hat{e}^d}). \tag{2.12}$$

This is the finite difference discretization of Laplacian used in the various elliptic operators throughout the algorithm, such as in (2.8).

The cell-centered projection can be constructed by wrapping two averaging operators around the MAC projection:

$$\mathbf{P} = \mathbf{I} - \mathbf{A}^{F\to C}(\mathbf{Q}^{mac}(\mathbf{A}^{C\to F})).$$

First, an operator to average cell-centered velocities to face centers is needed for the divergence in (2.8). For a face with a normal direction d, the averaging operation is

$$\mathbf{A}^{C\to F}(W^d)_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d} = \tfrac{1}{2}(W^d_{\boldsymbol{i}+\hat{e}^d} + W^d_{\boldsymbol{i}}). \tag{2.13}$$

Then, an averaging operator that is used to average gradients from face centers to cell centers is defined

$$\mathbf{A}^{F\to C}(\mathbf{G}^{mac}(\phi)^d)_{\boldsymbol{i}} = \tfrac{1}{2}(\mathbf{G}^{mac,d}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d}(\phi) + \mathbf{G}^{mac,d}_{\boldsymbol{i}-\frac{1}{2}\hat{e}^d}(\phi)). \tag{2.14}$$

The averaging operator, $\mathbf{A}^{F\to C}$, effectively results in a centered-difference for the gradient away from boundaries.

The face-centered gradient, $\mathbf{G}^{mac}$, of a cell-centered scalar, $\phi_{\boldsymbol{i}}$, is defined to be the finite difference approximation in the normal direction of the face:

$$\mathbf{G}^{mac}(\phi)^d_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d} = \frac{1}{h}(\phi_{\boldsymbol{i}+\hat{e}^d} - \phi_{\boldsymbol{i}}).$$

For homogeneous Neumann domain boundary conditions this gradient is 0; for Dirichlet domain boundary conditions, we use an odd extension of the solution at the boundary to obtain the gradient.

The transverse gradient at a face is comprised of the average of neighboring normal gradients in transverse directions, d', to a d-face:

$$\mathbf{G}^{mac}(\phi)^{d'}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d} = \frac{1}{N_{\mathcal{G}}} \sum_{\boldsymbol{i}+\frac{1}{2}\hat{e}^{d'}\in\mathcal{G}^{d',d}} (\mathbf{G}^{mac}(\phi)^{d'}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^{d'}})$$

where $\mathcal{G}^{d',d}$ is the set of faces in the transverse d' direction and $N_{\mathcal{G}}$ is the number of faces in this set. On a regular grid, $\mathcal{G}$ is the set of four neighboring adjacent faces in a d' direction. At solid wall domain boundaries, linear extrapolation of transverse gradients is used to preserve a constant pressure gradient as in Poiseuille flow. For transverse gradients whose face stencil crosses an orthogonal domain boundary the appropriate one-sided difference is taken.

MAC gradients which are pre-processed by the averaging operator, $\mathbf{A}^{F\to C}$, in (2.14) make use of linear extrapolation at boundary faces from interior faces to avoid over-specification of the problem. For the cell-centered projection target which is pre-processed by the averaging operator, $\mathbf{A}^{C\to F}$, in (2.13), boundary conditions are applied to the normal component. Referring to the channel boundary conditions (1.3)-(1.5), these are $W\cdot\mathbf{n} = u_{\text{in}}$ at the inlet, $W\cdot\mathbf{n} = 0$ at no-flow solid walls, and a quadratic extrapolation at the outlet that satisfies the Neumann boundary condition. Projection operator gradients are matched at coarse-fine interfaces by simple averaging as in [32].

2.3 Hyperbolic discretization

The advection term, $(U\cdot\nabla)U^{n+1/2}_{i,j}$, in (1.1) is discretized in conservation form since the flow is incompressible:

$$\nabla\cdot(UU)^{n+1/2}_{i,j} = \frac{1}{h}\left(u^{n+1/2}_{i+1/2,j}U^{n+1/2}_{i+1/2,j} - u^{n+1/2}_{i-1/2,j}U^{n+1/2}_{i-1/2,j} + v^{n+1/2}_{i,j+1/2}U^{n+1/2}_{i,j+1/2} - v^{n+1/2}_{i,j-1/2}U^{n+1/2}_{i,j-1/2}\right)$$

where i, j are the cell-centered grid indices in two dimensions, n is the number of the timestep, and $U = (u, v)$. Here, we consider two dimensions for ease of exposition—one direction that is normal to the flow (x), and one transverse (y)—with obvious extension of the transverse discretization to a third dimension (z).

We use an upstream-centered Taylor expansion to extrapolate the cell-centered velocity to the half step in time and cell edges:

$$\hat{U}^{n+1/2}_{i+1/2,j} = U^n_{i,j} + \frac{\Delta x}{2}\frac{\partial U^n}{\partial x} + \frac{\Delta t}{2}\frac{\partial U^n}{\partial t}.$$

Substitution of the PDE for the temporal derivative into the Taylor expansion yields extrapolated velocities in all directions from the cell center to both sides of a cell edge (or face, in 3D):

$$\begin{aligned}
\hat{U}^{x,+}_{i,j} &= U^n_{i,j} + \tfrac{1}{2}\min\left[\left(1 - u^n_{i,j}\frac{\Delta t}{\Delta x}\right), 1\right](\delta^N_x U)^n_{i,j} - \frac{\Delta t}{2\Delta y}v^n_{i,j}(\delta^T_y U)^n_{i,j} + \frac{\nu\Delta t}{2}\Delta U^n_{i,j} \\
\hat{U}^{x,-}_{i,j} &= U^n_{i,j} - \tfrac{1}{2}\min\left[\left(1 + u^n_{i,j}\frac{\Delta t}{\Delta x}\right), 1\right](\delta^N_x U)^n_{i,j} - \frac{\Delta t}{2\Delta y}v^n_{i,j}(\delta^T_y U)^n_{i,j} + \frac{\nu\Delta t}{2}\Delta U^n_{i,j} \\
\hat{U}^{y,+}_{i,j} &= U^n_{i,j} + \tfrac{1}{2}\min\left[\left(1 - v^n_{i,j}\frac{\Delta t}{\Delta y}\right), 1\right](\delta^N_y U)^n_{i,j} - \frac{\Delta t}{2\Delta x}u^n_{i,j}(\delta^T_x U)^n_{i,j} + \frac{\nu\Delta t}{2}\Delta U^n_{i,j} \\
\hat{U}^{y,-}_{i,j} &= U^n_{i,j} - \tfrac{1}{2}\min\left[\left(1 + v^n_{i,j}\frac{\Delta t}{\Delta y}\right), 1\right](\delta^N_y U)^n_{i,j} - \frac{\Delta t}{2\Delta x}u^n_{i,j}(\delta^T_x U)^n_{i,j} + \frac{\nu\Delta t}{2}\Delta U^n_{i,j}
\end{aligned}$$

where superscripts x and y refer to the coordinate direction of the extrapolation, and "+" and "−" indicate the direction of the extrapolation from the cell-center to the inside of an edge (inside left/bottom of an edge is the "+" state, inside right/bottom is the "−" state).

The monotonized 2nd-order normal slopes with van Leer limiting [56,15] are

$$(\delta^N_x U)^n_{i,j} = \begin{cases} (\delta_x U)^{vL} & \text{if } (U^n_{i+1,j} - U^n_{i,j})(U^n_{i,j} - U^n_{i-1,j}) > 0 \\ 0 & \text{if } (U^n_{i+1,j} - U^n_{i,j})(U^n_{i,j} - U^n_{i-1,j}) \le 0 \end{cases}$$

where

$$\begin{aligned}
(\delta_x U)^{vL} = &\text{sign }(U^n_{i+1,j} - U^n_{i-1,j}) \times \\
&\min(2|U^n_{i,j} - U^n_{i-1,j}|, 2|U^n_{i+1,j} - U^n_{i,j}|, \tfrac{1}{2}|U^n_{i+1,j} - U^n_{i-1,j}|).
\end{aligned}$$

The upwinded transverse slopes are

$$(\delta_y^T U)_{i,j}^n = \begin{cases} U_{i,j+1}^n - U_{i,j}^n + \frac{\nu\Delta t}{2}(\Delta U_{i,j+1}^n - \Delta U_{i,j}^n) \text{ if } v_{i,j}^n < 0 \\ U_{i,j}^n - U_{i,j-1}^n + \frac{\nu\Delta t}{2}(\Delta U_{i,j}^n - \Delta U_{i,j-1}^n) \text{ if } v_{i,j}^n \geq 0 \end{cases}$$

$$(\delta_x^T U)_{i,j}^n = \begin{cases} U_{i+1,j}^n - U_{i,j}^n + \frac{\nu\Delta t}{2}(\Delta U_{i+1,j}^n - \Delta U_{i,j}^n) \text{ if } u_{i,j}^n < 0 \\ U_{i,j}^n - U_{i-1,j}^n + \frac{\nu\Delta t}{2}(\Delta U_{i,j}^n - \Delta U_{i-1,j}^n) \text{ if } u_{i,j}^n \geq 0 \end{cases}$$

with a stability correction due to [37]. All slopes make use of one-sided differences at domain boundaries; at coarse-fine boundaries, we use linear interpolation and flux matching [33].

A Riemann problem is then solved to obtain the edge states, $\bar{U}$. For example, x-face states are

$$\bar{U}_{i+1/2,j} = \begin{cases} \hat{U}_{i,j}^{x,+} & \text{if } \frac{1}{2}(u_{i,j}^n + u_{i+1,j}^n) > 0 \\ \hat{U}_{i+1,j}^{x,-} & \text{if } \frac{1}{2}(u_{i,j}^n + u_{i+1,j}^n) < 0 \\ \frac{1}{2}(\hat{U}_{i,j}^{x,+} + \hat{U}_{i+1,j}^{x,-}) & \text{if } \frac{1}{2}(u_{i,j}^n + u_{i+1,j}^n) = 0. \end{cases}$$

To make up for the omitted pressure gradient in the velocity extrapolation, the solution to the Riemann problem is projected onto the space of divergence-free vectors using a MAC projection

$$U^{n+1/2} = \mathbf{P}^{mac}(\bar{U}) = \bar{U} - \mathbf{G}^{mac}((\mathbf{D}\mathbf{G}^{mac})^{-1}\mathbf{D}(\bar{U})). \tag{2.15}$$

The divergence is calculated as

$$\mathbf{D}(\bar{U}^{n+1/2})_{i,j} = \Big[\left(\bar{u}_{i+1/2,j} - \bar{u}_{i-1/2,j}\right) + \left(\bar{v}_{i,j+1/2} - \bar{v}_{i,j-1/2}\right) \Big]/h.$$

Both components of velocity have been accounted for up to this point, including the boundary conditions for the normal component. The transverse component of velocity at domain boundaries is taken to be the "+" or "−" state on the inside of the boundary edge, in keeping with the idea of an inviscid predictor step.

Our method has a stability constraint on the timestep due to the CFL condition for the advection terms:

$$\Delta t < \frac{\sigma h}{u^{\max}} \tag{2.16}$$

where $\sigma \leq 1$, and $u^{\max}$ is the magnitude of the maximum local wavespeed. For adaptive calculations, all levels of refinement use the same timestep. We note that subcycling in time is possible; however, it requires solution to an additional Poisson equation to enforce the divergence-free constraint with free-stream preservation [33].

3 Finite volume embedded boundary method

In grid cells where the irregular domain intersects the Cartesian grid, finite volume discretizations must be constructed to obtain conservative discretizations of flux-based operations defined by finite differences in the previous section. First, the underlying description of space is given by rectangular control volumes on a Cartesian grid $\Upsilon_{\boldsymbol{i}} = [(\boldsymbol{i} - \frac{1}{2}\boldsymbol{V})h, (\boldsymbol{i} + \frac{1}{2}\boldsymbol{V})h], \boldsymbol{i} \in \mathbb{Z}^D$, where D is the dimensionality of the problem, h is the mesh spacing, and $\boldsymbol{V}$ is the vector whose entries are all one. Given an irregular domain Ω, we obtain control volumes $V_{\boldsymbol{i}} = \Upsilon_{\boldsymbol{i}} \bigcap \Omega$ and faces $A_{\boldsymbol{i} \pm \frac{1}{2}\hat{e}^d}$ which are the intersection of the boundary of $\partial V_{\boldsymbol{i}}$ with the coordinate planes $\{\vec{x} : x_d = (i_d \pm \frac{1}{2})h\}$. The intersection of the boundary of the irregular domain with the Cartesian control volume is defined as $A^B_{\boldsymbol{i}} = \partial\Omega \bigcap \Upsilon_{\boldsymbol{i}}$. For ease of exposition, it is assumed that there is only one control volume per Cartesian cell. However, the algorithm described here has been generalized to allow for boundaries whose width is less than the mesh spacing, i.e., multi-valued cells. In regular cells, the finite volume approximation reduces to the finite difference method described in §2.

To construct finite volume methods using this description, several quantities need to be derived from the geometric objects:

- volume fractions, κ, and area fractions, α,

$$\kappa_{\boldsymbol{i}} = \frac{|V_{\boldsymbol{i}}|}{h^D}, \quad \alpha_{\boldsymbol{i}+\frac{1}{2}\hat{e}_s} = \frac{|A_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d_s}|}{h^{(D-1)}}, \quad \alpha^B_{\boldsymbol{i}} = \frac{|A^B_{\boldsymbol{i}}|}{h^{D-1}},$$

- centroids of the faces and of $A^B_{\boldsymbol{i}}$; and $\hat{n}_{\boldsymbol{i}}$, the average of outward normal of $\partial\Omega$ over $A^B_{\boldsymbol{i}}$,

$$\vec{x}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d} = \left[\frac{1}{|A_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d}|}\int_{A_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d}} \vec{x} dA\right]$$

$$\vec{x}^B_{\boldsymbol{i}} = \left[\frac{1}{|A^B_{\boldsymbol{i}}|}\int_{A^B_{\boldsymbol{i}}} \vec{x} dA\right]$$

$$\hat{n}_{\boldsymbol{i}} = \frac{1}{|A^B_{\boldsymbol{i}}|}\int_{A^B_{\boldsymbol{i}}} \hat{n} dA$$

where D is the dimension of space and $1 \leq d \leq D$. We assume we can compute all derived quantities to $O(h^2)$.

Geometric objects are determined by a hierarchical application of the divergence theorem to discrete values of implicit function representations of the geometry on the grid (see [36,31,47] for details on embedded boundary grid generation). Coarsened geometries are obtained by coarsening of the graph. The volume of a coarse cell is exactly the volume of the fine cells of which it is comprised. This grid generation machinery is part of the Chombo software libraries.

The conservative approximation of the divergence of a flux $\vec{F}$ can now be defined by applying a discrete form of the divergence theorem

$$\boldsymbol{D}(\vec{F})_{\boldsymbol{v}} = \frac{1}{h\kappa_{\boldsymbol{v}}}((\sum_{d=1}^{D}(\alpha_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d}\widetilde{F^d}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d} - \alpha_{\boldsymbol{i}-\frac{1}{2}\hat{e}^d}\widetilde{F^d}_{\boldsymbol{i}-\frac{1}{2}\hat{e}^d}) + \alpha^B_{\boldsymbol{v}}F^B_{\boldsymbol{v}}) \tag{3.17}$$

where $\widetilde{F}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^d}$ indicates that the flux has been interpolated to the face centroid using linear (2D) or bilinear (3D) interpolation of face-centered fluxes. For example, given the cell edge with outward normal $\hat{e}^1$, with centroid $\vec{x}$, the 2D linearly interpolated flux in the d $(d \neq 1)$ direction is defined by

$$\begin{aligned}
\widetilde{F^d}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^1} &= \eta F_{\boldsymbol{i}+\frac{1}{2}\hat{e}^1} + (1-\eta)F_{\boldsymbol{i}+\frac{1}{2}\hat{e}^1\pm\hat{e}^d} \\
\eta &= 1 - \frac{|\vec{x}\cdot\hat{e}^d|}{h_d} \\
\pm &= \begin{cases} + & \vec{x}\cdot\hat{e}^d > 0 \\ - & \vec{x}\cdot\hat{e}^d \leq 0. \end{cases}
\end{aligned} \tag{3.18}$$

The 3D bilinear interpolation of the flux for a face with normal $\hat{e}^1$ can be written as follows:

$$\begin{aligned}
\widetilde{F}_{\boldsymbol{i}+\frac{1}{2}\hat{e}^1} &= \omega F^d_{\boldsymbol{i}+\frac{1}{2}\hat{e}^1} + (1-\omega)F^d_{\boldsymbol{i}\pm\hat{e}^{d'}+\frac{1}{2}\hat{e}^1} \\
\omega &= 1 - \frac{|\vec{x}\cdot\hat{e}^{d'}|}{h_{d'}} \\
\pm &= \begin{cases} + & \vec{x}\cdot\hat{e}^{d'} > 0 \\ - & \vec{x}\cdot\hat{e}^{d'} \leq 0 \end{cases}
\end{aligned} \tag{3.19}$$

where $d' \neq d$, $d' \neq 1$ (see Figure 2).

3.1 Elliptic operators

The conservative discretization of the divergence theorem in equation (3.17) provides a flux-based formula for the discretization of the elliptic operations in the algorithm. We use the geometric multigrid approach described in [54] to solve these elliptic systems. In the context of Poisson's equation, as in the projections (2.8) and (2.15), the operator is the Laplacian and the flux is simply the gradient of a scalar, $\vec{F} = \nabla\varphi$, with Neumann boundary conditions, $F^B = \hat{n}\cdot\nabla\varphi = 0$, at the embedded boundary. However, in the context of Helmholtz, as in Eq. (2.6), the embedded boundary is a no-slip boundary for the velocity, requiring an elliptic operator with Dirichlet boundary conditions at the embedded boundary. In this case, the flux at the embedded boundary, $F^B = \hat{n}\cdot\nabla\varphi$, must be constructed while maintaining global 2^{nd}-order accuracy.

In [21], the flux at the embedded boundary due to a Dirichlet boundary condition is constructed by effectively casting a ray from the centroid of the boundary along the normal into the domain, interpolating φ to points along the ray (quadratic in 2D, biquadratic in 3D), computing the normal gradient of φ by differencing the interpolated points in 1D along the ray and obtaining a $\tau \approx O(h^2)$ local truncation error approximation of $\hat{n} \cdot \nabla\varphi$. In general, local truncation error on the interior of a domain is $\tau \approx O(h^2)$, and at the boundary $\tau \approx O(h/\kappa)$. It can be shown that global solution error is $\varepsilon = O(h^2)$. For the case of Dirichlet boundary conditions, the same conclusion holds for $\tau = O(1)$ at the boundary owing to the two orders of magnitude of freedom in the local truncation error resulting from the method of images (odd extensions) and the homogeneous condition at the boundary. (For Neumann boundary conditions, the minimum requirement to maintain second-order global error is $\tau = O(h)$.) The conclusion for Dirichlet boundary conditions, shown in [21] using potential theory, is that it is sufficient to have $O(1)$ boundary conditions to achieve second-order convergence of solution error for elliptic equations.

In practice, however, we have found that the 2nd-order stencil for Dirichlet boundary conditions first described in [21] is not stable for lower Reynolds number flows (much less the Stokes limit) and flows where there exist steep gradients near the boundary. To fix this instability, we make use of the two orders of magnitude of freedom in the local truncation error and instead apply a lower-order truncation error stencil ($\tau \approx O(h)$) to interpolate the flux at the irregular boundary centroid, $\mathbf{B}$ [51,53,52,48]. The flux, $\hat{n} \cdot \nabla\varphi$, is obtained by solving a least squares linear system for $\nabla\varphi$:

$$
\begin{aligned}
\mathcal{A} \cdot \nabla\varphi &= \delta\varphi \\
\mathcal{A} &= (\delta\vec{x}_1, \delta\vec{x}_2, ..., \delta\vec{x}_p)^T \\
\delta\varphi &= (\delta\varphi_1, \delta\varphi_2, ..., \delta\varphi_p)^T \\
\delta\vec{x}_m &= \vec{x}_m - \vec{x}_B \\
\delta\varphi_m &= \varphi_m - \varphi_B.
\end{aligned}
$$

The stencil of points ($m = 1, 2, ..., p$), which excludes the cut cell that contains the embedded boundary, is determined by the direction of the normal at the boundary. In 2D, the normal points to a quadrant which includes up, side and corner cells with $p = 3$, resulting in two equations and three unknowns in the least squares system. In 3D, the normal points to an octant with $p = 7$, resulting in three equations and seven unknowns. The stencils are shown in Figure 3.

In the case of very complex geometries such as those experienced in porous media flows where boundaries are very close together and can exhibit cusps and semi-disconnected cavities, this least squares stencil approach based on direction of the normal can be relaxed to use any points available in a monotone path from the root cell with a radius greater than 1 but with the same restrictions on p. For adaptive calculations, we use higher-order (quadratic) interpolation to fill ghost cells for 2nd-order elliptic operators (Laplacian) at coarse-fine boundaries in order to avoid $O(1)$ truncation error [32].

We also note that geometric multigrid coarsening can be challenged in very complex geometries. As an example, for the pressure-Poisson equations (2.8) and (2.15), the

presence of a semi-disconnected cavity in the domain can result in a Neumann problem with non-zero null space. We, therefore, rely on a combined embedded boundary-algebraic multigrid (EB-AMG) approach to solve elliptic equations in very complex geometry cases [49].

3.2 Advective derivative

Since the flow is incompressible, we make use of the conservative form of the advection term, $\nabla \cdot (\vec{u}\vec{u})$, in (3.17) with $\mathbf{F} = \vec{u}\vec{u}$. The problem with this discretization for advection is that the CFL stability constraint on the timestep is at best $\Delta t = O(\frac{h}{v_{\boldsymbol{i}}^{max}}(\kappa_{\boldsymbol{i}})^{\frac{1}{D}})$, where $v_{\boldsymbol{i}}^{max}$ is the magnitude of the maximum wave speed for the $\boldsymbol{i}^{th}$ control volume. This is the well-known small-cell problem for embedded boundary, cut cell methods. There have been a number of proposals to deal with this problem, including merging the small control volumes with nearby larger ones [46,14,45,27], the development of specialized stencils that guarantee the required cancellations [10,7,20,25], or simply using a threshold volume below which the cell is considered completely covered, i.e., $\kappa = 0$, as in [17].

Our previous approach in [54] was to expand the range of influence of the small control volumes algebraically to obtain a stable method, akin to [11,6,42]. We used a linear hybridization of conservative and non-conservative estimates of $\nabla \cdot (\vec{u}\vec{u})$:

$$\nabla \cdot (\vec{u}\vec{u})_{\boldsymbol{i}}^{n+1/2} = \kappa_{\boldsymbol{i}}(\nabla \cdot (\vec{u}\vec{u}))_{\boldsymbol{i}}^{C} + (1-\kappa_{\boldsymbol{i}})(\nabla \cdot (\vec{u}\vec{u}))_{\boldsymbol{i}}^{NC}).$$

The small denominator in $\nabla \cdot (\vec{u}\vec{u})$ is canceled, and a stable method is obtained. However, the method fails to conserve mass by an amount measured by the difference between the hybrid discretization and the conservative one:

$$\delta M_{\boldsymbol{i}} = \kappa_{\boldsymbol{i}}((\nabla \cdot (\vec{u}\vec{u}))_{\boldsymbol{i}}^{C} - (\nabla \cdot (\vec{u}\vec{u}))_{\boldsymbol{i}}^{NC}) = \kappa_{\boldsymbol{i}}(1-\kappa_{\boldsymbol{i}})(\nabla \cdot (\vec{u}\vec{u}))_{\boldsymbol{i}}^{C} - (\nabla \cdot (\vec{u}\vec{u}))_{\boldsymbol{i}}^{NC}).$$

To maintain overall conservation, $\delta M_{\boldsymbol{i}}$ can be redistributed into nearby cells $\boldsymbol{i}'$,

$$\nabla \cdot (\vec{u}\vec{u})_{\boldsymbol{i}'}^{n+1/2} := \nabla \cdot (\vec{u}\vec{u})_{\boldsymbol{i}'}^{n+1/2} + w_{\boldsymbol{i},\boldsymbol{i}'}\delta M_{\boldsymbol{i}} \ , \ \boldsymbol{i}' \in N(\boldsymbol{i}),$$

$$w_{\boldsymbol{i},\boldsymbol{i}'} \geq 0, \ \sum_{\boldsymbol{i}' \in N(\boldsymbol{i})} w_{\boldsymbol{i},\boldsymbol{i}'}\kappa_{\boldsymbol{i}'} = 1 \tag{3.20}$$

where $N(\boldsymbol{i})$ is some set of indices in the neighborhood of $\boldsymbol{i}$, and including $\boldsymbol{i}$. The sum condition (3.20) makes the redistribution step conservative. The weights $w_{\boldsymbol{i},\boldsymbol{i}'}$ must be bounded independent of $(\kappa_{\boldsymbol{i}'})^{-1}$. We use volume weighted redistribution,

$$w_{\boldsymbol{i},\boldsymbol{i}'} = (\sum_{\boldsymbol{i}' \in N(\boldsymbol{i})} \kappa_{\boldsymbol{i}'})^{-1}$$

where $N(\boldsymbol{i})$ is a set of indices, including $\boldsymbol{i}$, within a radius of influence of one and connected by a monotone path.

The success of this approach depends on the calculation of $\nabla \cdot (\vec{u}\vec{u})^{NC}$ because it is almost entirely responsible for the update of $\nabla \cdot (\vec{u}\vec{u})_{\boldsymbol{i}}$ in control volumes with

$\kappa_{\boldsymbol{i}} \ll 1$. Specifically, $\nabla \cdot (\vec{u}\vec{u})^{NC}$ must be designed so that the solution in small control volumes comes into equilibrium with the larger control volumes around it. We now enforce this point by summing the conservative approximation itself in a domain of influence around the cut cell and normalizing it by the sum of volume fractions in those cells to obtain the non-conservative approximation:

$$\nabla \cdot (\vec{u}\vec{u})^{NC}_{\boldsymbol{i}} = \frac{\sum\limits_{\boldsymbol{i}' \in N(\boldsymbol{i})} (\kappa_{\boldsymbol{i}'} \nabla \cdot (\vec{u}\vec{u})^{C}_{\boldsymbol{i}'})}{\sum\limits_{\boldsymbol{i}' \in N(\boldsymbol{i})} \kappa_{\boldsymbol{i}'}}. \tag{3.21}$$

To compute $\nabla \cdot (\vec{u}\vec{u})^{C}$, fluxes $\vec{u}\vec{u}$ are interpolated to face centroids as in (3.18) or (3.19), and substituted into (3.17).

3.3 Volume-weighted scheme

In a new approach we avoid the small cell problem altogether by taking advantage of the structure of our finite volume elliptic solvers which take the form $\kappa\mathcal{L} = \kappa\rho$ where $\mathcal{L}$ is the elliptic operator, ρ is the right-hand side and κ is the volume fraction of a cut cell. This volume-weighted form allows us to compute source terms in (2.6) which are also volume-weighted. We also introduce a conservative form of the cell-centered pressure gradient in the pressure-correction form of the projection. The overall algorithm is as follows:

(1) Initially a cell-centered velocity is obtained from the projection of the prescribed conditions, $U^0 = \mathbf{P}(U^{\text{init}})$, similar to a potential flow solution. The pressure gradient is constructed to balance the viscous stress from this initial velocity, $\nabla p^{-1/2} = \nu \Delta U^0$, to ensure a stable calculation (see §4.3 for details) and then made to be volume-weighted, $\kappa \nabla p^{-1/2}$.

(2) If the flow is inertial (say, $Re > 0.1$), then velocities are extrapolated from cell centers to cell edges as in §2.3 and only the conservative volume weighted advection term is computed, $\kappa(\nabla \cdot (UU)^{n+1/2})$. If the Reynolds number is low (say, $Re < 0.1$), or approaches the Stokes limit such that $\vec{u} \cdot \nabla \vec{u} \ll \nu \Delta \vec{u}$, then this step is unnecessary.

(3) The implicit Helmholtz equation (2.6) is solved in the form of our finite volume elliptic equation $\kappa\mathcal{L} = \kappa\rho$ where the right-hand side has volume-weighted terms including the source term:

$$\kappa(I - \frac{\nu\Delta t}{2}\Delta)U^{n+1,*} = \kappa(U^n + \Delta t(\frac{\nu\Delta t}{2}\Delta U^n - (U \cdot \nabla)U^{n+1/2} - \nabla p^{n-1/2})). \tag{3.22}$$

(4) For the approximate projection a volume-weighted Poisson equation is solved

$$\kappa\Delta\delta = \kappa\mathbf{D}(U^{n+1,*}) \tag{3.23}$$

where $\delta = p^{n+1/2} - p^{n-1/2}$ indicates pressure correction form.

(5) The volume-weighted cell-centered gradient of the pressure correction, δ, is computed using a corollary to the divergence theorem for gradients

$$\kappa \mathbf{G}(\delta) = V \iiint \frac{\partial \delta}{\partial x_i} dV = V \iint \delta(\hat{e}_i \cdot \hat{n}_i) dA. \tag{3.24}$$

The value of the pressure correction at the cell center can be used at the boundary centroid or a more elaborate least squares system can be solved for the boundary value.

(6) The volume-weighted intermediate velocity is corrected with the volume-weighted gradient of the pressure correction

$$\kappa U^{n+1} = \kappa U^{n+1,*} - \kappa \mathbf{G}(\delta). \tag{3.25}$$

(7) If the flow is inertial, then the volume weighting is removed from velocity in a normalization procedure similar to (3.21) so that the velocity can be used in the advection step of the next timestep

$$U^{n+1} = \frac{\sum\limits_{\boldsymbol{i}' \in N(\boldsymbol{i})} (\kappa_{\boldsymbol{i}'} U^{n+1}_{\boldsymbol{i}'})}{\sum\limits_{\boldsymbol{i}' \in N(\boldsymbol{i})} \kappa_{\boldsymbol{i}'}}. \tag{3.26}$$

Here, we note that normalization of the volume-weighted velocity is allowed because the velocity has already been sufficiently smoothed in the solution to the viscous Helmholtz equation (3.22). For consistency with step (2), this final step (7) is not necessary for low Reynolds number or Stokes flow.

4 Results

4.1 Accuracy

To demonstrate the accuracy of the algorithm we consider incompressible flow inside a sphere. The fluid is initialized as a Gaussian vortex

$$\omega(r) = e^{-20(4r-0.5)^2}$$

where r is measured from the center of the sphere at $\mathbf{x}_0$ to a point $\mathbf{x}$ as $r^2 = (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2$ in 3D. The initial velocity can then be prescribed by

$$\begin{aligned} u(\mathbf{x}, t_0) &= \omega(r)((z - z_0) - (y - y_0))/r \\ v(\mathbf{x}, t_0) &= \omega(r)((x - x_0) - (z - z_0))/r \\ w(\mathbf{x}, t_0) &= \omega(r)((y - y_0) - (x - x_0))/r. \end{aligned}$$

In 2D, $r^2 = (x - x_0)^2 + (y - y_0)^2$ and

$$u(\mathbf{x}, t_0) = -\omega(r)(y - y_0)/r$$
$$v(\mathbf{x}, t_0) = \quad \omega(r)(x - x_0)/r.$$

The Reynolds number for this study is $Re = 5$ based on vortex strength and diameter.

We estimate the error in the solution using the standard Richardson procedure where computations of differing resolutions are evolved to the same time and compared in a certain norm (see [36] for details). Convergence rates for the 2D state variables are displayed in Tables 1, 2 and 3 for L^1, L^2 and L^∞ norms, respectively. We demonstrate 2^{nd}-order accuracy for all variables in all norms. A fixed step size of $\Delta t = 0.00025$, which is equivalent to a CFL number of $\sigma = 0.5$, is run for 512 steps at the finest resolution of $h = 1/2048$. We note that the pressure gradient resulting from a single application of the projection is 1^{st}-order (the scalar pressure is 2^{nd}-order) [16,50]. To obtain a 2^{nd}-order pressure gradient, an additional approximate projection is required. The computational cost of this additional projection is minimized by initialization of the pressure to the value obtained from the first application of the projection.

We show convergence results in 3D in Tables 4, 5 and 6. At the finest resolution of $h = 1/256$, a fixed step size of $\Delta t = 0.0005$, which is equivalent to a CFL number of $\sigma = 0.5$, is run for 64 steps. The convergence rates are 2^{nd}-order for all variables in all norms, except that the velocity components are slightly less than 2^{nd}-order in the L^∞-norm. We attribute this slight degradation in accuracy to the solution not being fully resolved in the asymptotic regime for convergence.

4.2 Stability of the approximate projection

We demonstrate that the approximate projection operator is stable, i.e., $||\mathbf{P}|| < 1$ (see [29]), by showing that the divergence of a velocity field diminishes with repeated application of the projection. The velocity field is initialized to a potential flow past an infinitely long cylinder with radius = 0.1. The cylinder is in the center of a unit square domain. We iteratively project the velocity field, U, and evaluate the norm of the divergence, $\kappa\mathbf{D}(U)$, and the norm of the pressure gradient, $\nabla\phi$, after each projection. Figures 4–7 show that all norms of both fields monotonically decrease with number of projection iterations. Flattening of the curves near the end is due to the residual of the solution to the Poisson equation by multigrid iterations approaching machine accuracy.

4.3 Stability in the Stokes limit

We use the algorithm to compute a range of unsteady flows, including low Reynolds number flows where a steady-state may exist. In this flow regime, parameters can approach the Stokes limit, $Re \to 0$, where $Re = U_c l_c/\nu$ and U_c, l_c are characteristic quantities, leading to the Stokes equations (Navier-Stokes less the advective derivative). However, the Stokes equations do not capture all the physics of flows even at

$Re = 0.1$, a value that has traditionally been considered well inside the Stokes limit. We demonstrate this point by considering flow near a sharp corner, as in [38]. Figure 8 shows a comparison between solving the unsteady Stokes and Navier-Stokes equations to a steady-state in an abrupt expansion channel, both for the same initial data at $Re = 0.1$. The magnitude of the velocity and extent of the recirculation zone are noticeably greater when $\vec{u} \cdot \nabla \vec{u}$ is included in the calculation as seen in the difference in the location of the innermost contour. Comparison of plots of the velocity along a line through the recirculation zone shows a 5% difference. The difference between the solutions is more dramatic as the Reynolds number increases. The criterion for a steady-state solution is $U^{n+1} - U^n < \epsilon \Delta t$, where $\epsilon = 10^{-8}$.

In order to perform stable computations for low Reynolds number flows ($Re < 0.1$), we must construct a well-posed IBVP such that both the momentum and continuity equations are satisfied by the initial conditions. We obtain a divergence-free potential flow field that satisfies the no-flow normal boundary conditions by projecting the velocity: $U^0 = \mathbf{P}(U^{\text{init}})$, where $U^{\text{init}} = 0$ inside the domain. The pressure gradient is calculated to balance the viscous stress due to the flow field: $\nabla p^{-1/2} = \nu \Delta U^0$. In this regime, since viscous effects can dominate inertial forces ($\nu > U_c l_c$), we define the viscous timestep to be $\Delta t_\nu = h^2/\nu$. The stability constraint for the algorithm in the low Reynolds number regime is $\Delta t = \min(\Delta t_{\text{CFL}}, \Delta t_\nu)$.

4.4 Scalability and performance

The algorithm described here has been implemented in the Chombo software framework. Chombo provides a set of tools for implementing finite difference and finite volume methods for the solution of PDEs on block-structured adaptively refined rectangular grids. Chombo also supports computations in complex geometries with embedded boundaries. Chombo software libraries enable high performance computing, data management and I/O for large scale simulations.

We demonstrate the scalability and performance of our Chombo-based algorithm using a weak scaling test for flow through a cylinder packed with spheres (see Figure 10) as in [49]. In weak scaling the problem domain is refined by the same factor as the increase in the number of processor cores (e.g., factor of 2 refinement in each spatial dimension, D, requires $2^D\times$ the number of cores). These tests are conducted on the NERSC Cray XC30 system, Edison, for up to 131,072 cores and on the OLCF Cray XK7 system, Titan, for up to 262,144 CPU cores. We perform 10 timesteps of the algorithm and take the average time per timestep in seconds. We use a sweet spot for domain decomposition and load balancing of one box per processor core where one box is 32^3 cells. Since a large number of spheres have to be randomly placed in a cylinder, it is difficult to guarantee a fixed number of spheres per box. However, the scaling is theoretically very close to replicated data as in [54]. Therefore, we take three different aspect ratios of the cylinder—where each aspect ratio is a weak scaling test in itself—and combine into one continuous scaling curve in Figure 9. The three sets of weak scaling data are depicted by shape: a 1-to-1 cylinder packed with 750 spheres run on 512-4096-32768-262144 cores (squares), a 2-to-1 cylinder with 1500

spheres run on 1024-8192-65536 (triangles) and a 4-to-1 cylinder with 3000 spheres run on 2048-16384-131072 cores (circles).

On Edison, we observe excellent performance with about 83% efficiency from 512 to 131,072 cores—a relatively flat weak scaling curve—and an average time per timestep of 20 seconds at the highest concurrency. On Titan, we observe about 67% efficiency from 512 to 131,072 cores, and only 50% up to 262,144 cores. We note a slight dip at $N = 4096$, and even slighter at $N = 32768$, in both curves that is likely due to a lower percentage of cut cells from refinement of the geometry. We do observe an upward trend in the weak scaling curve on Titan, particularly at the two highest concurrencies (it should be flat throughout), but overall the time only slightly doubles from the lowest concurrency to the highest. We consider the result on Titan to be good performance for the vast range of concurrencies, and given the flow physics and geometry. Furthermore, performance is in the neighborhood of 1 timestep per minute at the highest concurrency on Titan, which is an acceptable metric for a large scale fluid dynamics calculation. We do not make use of the GPUs on Titan for this scaling test, which may contribute to degraded performance.

4.5 Simulation results

We present simulation results for the 2D and 3D incompressible Navier-Stokes equations for a range of Reynolds numbers in various geometries. Flow problems are set up such that the flow is typically from left to right in the x-direction, the kinematic viscosity is that of water, $\nu = 0.01$ cm^2/sec, the average velocity at inflow is 1 cm/sec (Poiseuille in 2D, constant in 3D), unless otherwise stated. For flows where inertial forces have an effect (typically $Re > 0.1$), the CFL number is $\sigma = 0.9$. All units are specified in the CGS system. The maximum grid size resulting from domain decomposition and the AMR hierarchy is 256^2 cells per grid block (box) in 2D and 32^3 cells per grid block in 3D. The criterion for steady-state flow is $U^{n+1} - U^n < \epsilon \Delta t$, where $\epsilon = 10^{-8}$.

4.5.1 Low Reynolds number flow

We demonstrate the algorithm at the low end of the Reynolds number flow regime by showing steady-state results for the packed cylinder used in the scaling study. Using 65,536 processor cores on the NERSC Cray XC30 Edison we simulate steady-state flow in the 2-to-1 cylinder packed with 1500 spheres, as in Figure 10b. In Figures 11 and 12 we show the axial (x) and transverse (z) velocities with magnified views to convey the tortuosity of the flow and the resolved viscous boundary layer in the pore space. The grid resolution for this simulation is $2048 \times 1024 \times 1024$ cells.

To demonstrate hero run capability we also simulated steady-state flow in the 4-to-1 cylinder geometry packed with 3000 spheres in Figure 10c. The steady-state velocity is shown in Figure 13. This simulation made use of 131,072 processor cores on OLCF Titan. The Reynolds number is 0.5 for both simulations.

4.5.2 Direct numerical simulation from image data.

In addition to synthetic geometries as in the packed cylinder we demonstrate direct simulation from microtomography image data. Figure 14 shows flow in Bedford limestone with porosity of 29%. In this simulation, resolution of the cross section is critical in order to capture viscous effects in very tight pore space. The grid resolution of this simulation is $2048 \times 2048 \times 320$ cells, or $h = 43$nm. The image voxel size is 4.4 μm.

We also demonstrate the ability to model fully resolved steady-state flow in a fractured shale in Figure 15. The geometry in this case is obtained from focused ion beam scanning electron microscopy (FIB-SEM) image data. The pore space is tighter than the limestone, even though a fracture aperture is present, with a porosity of 18%. The grid resolution is $1920 \times 1600 \times 640$, or $h = 48$nm. The image voxel size is 50 nm. In both the synthetic packed cylinder and the image data simulations we make use of the EB-AMG method in [49] to solve elliptic problems in these very complex geometries.

4.5.3 Flow past a cylinder ($Re = 300$)

We perform direct numerical simulation (DNS) of flow past a cylinder in both 2D and 3D. The diameter of the cylinder centered at $x = 1$ is $d = .125$ in a domain that has dimensions $l = 16$ and $w = 8$. With $U_c = 1$, $l_c = d = 0.125$ and $\nu = 0.0004167$, the Reynolds number for this simulation is $Re = 300$. In Figure 16a we performed a highly resolved four-level AMR calculation in 2D at $Re = 300$ where the finest level covers only 5% of the total domain at $t = 98$. The length of wake in this calculation is a very long $120d$, which is shown to be necessary to capture the halfway downstream secondary structures and far downstream tertiary structures. (We have also simulated an extended domain with twice the length ($l = 32$) shown in Figure 19 that depicts additional wake structures, but with no comparison to 3D.) A recirculation zone persists within the secondary wake structure along the centerline between the vortices above and below. We simulate a domain width of $64d$ and use slip wall boundary conditions at boundaries transverse to the x-direction of the flow to minimize the interaction of domain boundary effects with the wake. The additional mesh refinement tracks dynamically with the magnitude of vorticity in time using a refinement threshold of 2.0. Grid blocks are outlined (cells not shown) for this five-level calculation in Figure 17 (top). Each grid block contains a maximum of 16^2 cells for a given level of refinement.

We performed a simulation in 3D for comparison to 2D using the same parameters, but with only two additional AMR levels. Williamson notes a transition Reynolds number regime up to 300, beyond which velocity fluctuations become irregular and vortex formation is three-dimensional [57–59]. In the 3D simulation, a number of transient structures develop in the fluid that organize at very long time into a persistent train of vortices (see Figure 16b,c), which is only similar to the near wake in 2D (see Figure 16a). The third dimension has a self-organizing effect on the wake structures as previously noted [24,61]. The flow in the wake is clearly not two-dimensional as seen in the bowing of the peaks and valleys, and narrowing of the length of the rows.

This point is further emphasized in the isocontour plot of z-vorticity in Figure 18. We also show the grid blocks with cells for this two-level calculation in Figure 17 (bottom). The finest level is gridded on 12% of the domain at $t = 98$. Each grid block contains a maximum of 16^3 cells for a given level of refinement.

4.5.4 Flow past a sphere ($Re = 600$)

We perform DNS of 3D flow past a sphere. The diameter of the sphere, centered at $x = 1$, is $d = .125$ in a domain that has dimensions $l = 16$ and $w = 8$. With $U_c = 2$, $l_c = d = 0.125$ and $\nu = 0.0004167$, the Reynolds number for this simulation is $Re = 600$. In Figure 20 we show a resolved three-level AMR calculation where the finest level covers less than 1% of the total domain. (The domain is very large in order to minimize boundary interactions with the wake.) The base grid is $512 \times 256 \times 256$ with two additional levels of refinement, factor of 4. The maximum grid box size in the AMR hierarchy is 32^3. The 3D plots at the top of Figure 20 depict a notched wake in the velocity field, but no oscillations at $t = 2$ seconds. We show outlines of the grid boxes to indicate refinement only around the sphere and the wake, and not away from the interesting part of the flow. The second row of plots shows the same in a 2D slice. At the bottom of the figure we show a transverse component of velocity as well as the pressure. Figure 21 depicts a wake that has begun to oscillate at $t = 2.5$ seconds.

4.5.5 High Reynolds number flow in a contraction ($Re = 6300$)

We show results for another example of the effectiveness of AMR when combined with the embedded boundary method by demonstrating DNS of flow in a sudden contraction. This example is intended to model flow of oil upward in a long (over 4 km) pipe buried in the sea bed that undergoes essentially a contraction at the sea floor near a blowout preventer (as in the Deepwater Horizon Macondo well [35,41]). In such a scenario it is critical to solve for the bulk flow characteristics like pressure drop and flow rate over the entire length of the pipe in order to assess the likelihood of success for intervention strategies. However, it is equally critical to resolve the microscopic, by comparison, boundary layer effects near the blowout preventer in order to be able to determine failure points. Here, we focus on Newtonian flow in a 1 meter length section of the pipe near a 4-to-1 contraction in two dimensions of Cartesian coordinates for demonstration purposes only. The base grid contains 2048×1024 cells (0.488 mm resolution) with 32 boxes of 256^2 cells. Three additional levels of refinement (factor of 4) are added dynamically for an effective resolution of 7.6 μm near the contraction. The finest AMR level contains 8975 boxes and covers less than 3% of the domain. The Reynolds number is approximately 6300. Figure 22 depicts transient turbulent flow in the contraction at $t = 0.125$ seconds. We show both velocity and pressure, with extreme gradients near and just downstream of the contraction. We show increased magnification in the lower figures, with box boundaries and, in the bottom row, mesh resolution.

5 Conclusions

We present a conservative, second-order accurate method to solve the incompressible Navier-Stokes equations in complex geometries. The method is based on a finite volume, embedded boundary approach that makes use of the discrete form of the divergence theorem to discretize the solution in irregular control volumes resulting from the intersection of solid boundaries with a regular, Cartesian grid. The method reduces to a standard finite difference approach in regular cells away from the boundary. We introduce several novel ideas including a volume-weighted scheme that avoids the small cell problem associated with cut cell methods and a conservative cell-centered gradient for approximate projections. We have coupled the embedded boundary method with AMR to provide a high-performance, high-resolution simulation tool for modeling multiscale, multiphysics problems in complex geometries. The algorithm scales to 262,144 processor cores and is amenable to direct simulation from image data. The cut cell algorithm described here is the basis for a high performance production code that models 3D engineering scale problems involving incompressible viscous flow and transport in complex geometries. We demonstrate the robustness of the algorithm for a wide range of Reynolds numbers and flow geometries—from creeping flow in realistic pore space to transitional flows past bluff bodies to turbulent pipe flow.

We model moderate Reynolds number phenomena for flow past a cylinder at a fidelity that has not yet been achieved. Typically, only the near wake of the cylinder is modeled numerically, as in [24,61]. In 2D, we observe secondary, tertiary and even quaternary structures far downstream of the near wake at a scale that is much broader than previously modeled, up to 250 cylinder diameters downstream in the wake. By comparison, in 3D, we observe a very long and persistent single train of vortices with coherent structures in the cross channel direction for the same length wake as in the 2D case. Similarly, we have also demonstrated high resolution of turbulent flow past a sphere in the early stages of wake formation. This capability could prove to be very effective in an investigation of both near and far wake dynamics in high Reynolds number flows past bluff bodies.

We have shown that the method is also suitable for direct numerical simulation of high Reynolds number internal flows. The adaptive capability captures small scale, microscopic features in the viscous boundary layer near a singular geometric feature such as a contraction while also resolving bulk flow properties in a domain that is 6 orders of magnitude larger than the finest spatial resolution of the boundary layer. This demonstration was motivated by a large scale engineering model for worst case discharge and failure point analysis of the Deepwater Horizon Macondo oil well blowout in 2010.

With demonstrated capability to perform direct simulation from image data, the algorithm has served as the basis for low Reynolds number reactive transport simulations in realistic pore space [39,40] and is proving to be a useful tool for modeling flow in fractured subsurface materials. The algorithm is also amenable to methods for tracking fluid-fluid and fluid-solid interfaces [36], providing a consistent approach to modeling multiphase flow and time-dependent boundary problems.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, and in part by the Office of Basic Energy Sciences Energy Frontier Research Centers and used resources of the National Energy Research Scientific Computing Center, all under contract number DE-AC02-05CH11231. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported by the Office of Science of the Department of Energy under contract number DE-AC05-00OR22725. This research used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357. Simulation data in Figure 14 is based upon synchrotron microtomography imagery acquired by Jonathan Ajo-Franklin and Marco Voltolini at the Advanced Light Source, Beamline 8.3.2, which is supported by the Office of Science, Office of Basic Energy Sciences, of the U.S. DOE under contract DE-AC02-05CH11231. Simulation data in Figure 15 is based upon FIB-SEM imagery obtained by Lisa Chan at Tescan USA and processed by Terry Ligocki (LBNL), courtesy of Tim Kneafsey (LBNL).

References

[1] M. J. Aftosmis, J. Melton, and M. J. Berger. Robust and efficient Cartesian mesh generation for component-base geometry. *AIAA Journal*, 36(6):952–960, June 1998.

[2] A. Almgren, J. Bell, P. Colella, and T. Marthaler. A Cartesian grid projection method for the incompressible Euler equations in complex geometries. *SIAM Journal on Scientific Computing*, 18(5):1289–1309, 1997.

[3] A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. J. Welcome. A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations. *J. Comput. Phys.*, 142(1):1–46, May 1998.

[4] J. B. Bell, P. Colella, and H. M. Glaz. A second-order projection method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 85:257–283, 1989.

[5] J. B. Bell, P. Colella, and L. H. Howell. An efficient second order projection method for viscous incompressible flow. In *AIAA 10th Comp. Fluid Dynamics Conf.*, pages 360–367, 1991.

[6] J. B. Bell, P. Colella, and M.L. Welcome. Conservative front-tracking for inviscid compressible flow. In *AIAA 10th Computational Fluid Dynamics Conference. Honolulu*, pages 814–822, 1991.

[7] M. Berger, C. Helzel, and R. LeVeque. H-box methods for the approximation of hyperbolic conservation laws on irregular grids. *SIAM Journal of Numerical Analysis*, 41:893–918, 2003.

[8] M. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, March 1984.

[9] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82(1):64–84, May 1989.

[10] M. J. Berger and R. J. LeVeque. Stable boundary conditions for Cartesian grid calculations. Technical Report 90-37, ICASE, May 1990.

[11] I. L. Chern and P. Colella. A conservative front-tracking method for hyperbolic conservation laws. Technical Report UCRL-97200, Lawrence Livermore National Laboratory, 1987.

[12] A. J. Chorin. Numerical solutions of the Navier-Stokes equations. *Math. Comp.*, 22:745–762, 1968.

[13] M-H. Chung. Cartesian cut cell approach for simulating incompressible flows with rigid bodies of arbitrary shape. *Computers & Fluids*, 35(6):607 – 623, 2006.

[14] W. J. Coirier and K. G. Powell. An assessment of Cartesian-mesh approaches for the Euler equations. *J. Comput. Phys.*, 117:121–131, 1995.

[15] P. Colella, D. T. Graves, B. Keen, and D. Modiano. A Cartesian grid embedded boundary method for hyperbolic conservation laws. *J. Comput. Phys.*, 211:347–366, 2006.

[16] P. Colella and D. P. Trebotich. Numerical simulation of incompressible viscous flow in deforming domains. *Proc Natl Acad Sci USA*, 96(10):53785381, 1999.

[17] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (The Ghost Fluid Method). *J. Comput. Phys.*, 152(2):457–492, 1999.

[18] R. Fedkiw and X-D. Liu. The Ghost Fluid Method for Viscous Flows. In M. Hafez and J.-J. Chattot, editors, *Innovative Methods for Numerical Solutions of Partial Differential Equations*, pages 111–143. World Scientific Publishing, New Jersey, 2002.

[19] D. T. Graves, P. Colella, D. Modiano, J. Johnson, B. Sjogreen, and X. Gao. A Cartesian grid embedded boundary method for the compressible NavierStokes equations. *Comm. App. Math. Comp. Sci.*, 8(1):99–122, 2013.

[20] C. Helzel, M. J. Berger, and R. J. LeVeque. A high–resolution rotated grid method for conservation laws with embedded geometries. *SIAM J. Sci. Stat. Comput.*, 2005.

[21] H. S. Johansen and P. Colella. A Cartesian grid embedded boundary method for Poisson's equation on irregular domains. *J. Comput. Phys.*, 147(2):60–85, December 1998.

[22] C. R. Ethier K. Shahbazi, P. F. Fischer. A high-order discontinuous Galerkin method for the unsteady incompressible NavierStokes equations. *J. Comput. Phys.*, 222(1):391–407, 2007.

[23] J. Van Kan. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM Journal on Scientific and Statistical Computing*, 7:870–891, 1986.

[24] G. E. Karniadakis and G. S. Triantafyllou. Three-dimensional dynamics and transition to turbulence in the wake of bluff objects. *J. Fluid Mech.*, 238:1–30, 1992.

[25] B. Keen and S. Karni. A second order kinetic scheme for gas dynamics on arbitrary grids. *J. Comput. Phys.*, 2005.

[26] J. Kim and P. Moin. Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.*, 59:308–323, 1985.

[27] M.P. Kirkpatrick, S.W. Armfield, and J.H. Kent. A representation of curved boundaries for the solution of the NavierStokes equations on a staggered three-dimensional Cartesian grid. *J. Comput. Phys.*, 184(1):1 – 36, 2003.

[28] M.-C. Lai and C. S Peskin. An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity . *J. Comput. Phys.*, 160:705–719, 2000.

[29] M. F. Lai. *A Projection Method for Reacting Flow in the Zero Mach Number Limit.* PhD thesis, University of California, Berkeley, 1994.

[30] R. LeVeque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis*, 31(4):1019–1044, 1994.

[31] T. J. Ligocki, P. O. Schwartz, J. Percelay, and P. Colella. Embedded boundary grid generation using the divergence theorem, implicit functions, and constructive solid geometry. *J. Phys: Conference Series*, 125(1):1–5, 2008.

[32] D Martin and P Colella. A cell-centered adaptive projection method for the incompressible Euler equations. *J. Comput. Phys.*, 2000.

[33] D. F. Martin, P. Colella, and D. T. Graves. A cell-centered adaptive projection method for the incompressible Navier-Stokes equations in three dimensions. *J. Comput. Phys.*, 227:1863–1886, 2008.

[34] P. McCorquodale, P. Colella, and H. Johansen. A Cartesian grid embedded boundary method for the heat equation on irregular domains. *J. Comput. Phys.*, 173:620–635, November 2001.

[35] M. K. McNutt, S. Chu, J. Lubchenco, T. Hunter, G. Dreyfus, S. A. Murawski, and D. M. Kennedy. Applications of science and engineering to quantify and control the Deepwater Horizon oil spill. *Proceedings of the National Academy of Sciences*, 109(50):20222–20228, 2012.

[36] G. H. Miller and D. Trebotich. An embedded boundary method for the Navier-Stokes equations on a time-dependent domain. *Comm. App. Math. Comp. Sci.*, 7(1):1–31, 2012.

[37] M. L. Minion. On the stability of Godunov-projection methods for incompressible flow. *J. Comput. Phys.*, 123(2):435–449, Feb. 1996.

[38] H. K. Moffatt. Viscous and resistive eddies near a sharp corner. *J. Fluid Mech.*, 18:1–18, 1964.

[39] S. Molins, D. Trebotich, C. I. Steefel, and C. Shen. An investigation of the effect of pore scale flow on average geochemical reaction rates using direct numerical simulation. *Water Resources Research*, 48(3):n/a–n/a, 2012.

[40] S. Molins, D. Trebotich, L. Yang, J. B. Ajo-Franklin, T. J. Ligocki, C. Shen, and C. I. Steefel. Pore-scale controls on calcite dissolution rates from flow-through laboratory and numerical experiments. *Environmental Science & Technology*, 48(13):7453–7460, 2014. PMID: 24865463.

[41] C. M. Oldenburg, B. M. Freifeld, K. Pruess, L. Pan, S. Finsterle, and G. J. Moridis. Numerical simulations of the Macondo well blowout reveal strong control of oil flow by reservoir permeability and exsolution of gas. *Proceedings of the National Academy of Sciences*, 109(50):20254–20259, 2012.

[42] R. B. Pember, J. B. Bell, P. Colella, W. Y. Crutchfield, and M. L. Welcome. An adaptive Cartesian grid method for unsteady compressible flow in irregular regions. *J. Comput. Phys.*, 120(2):278–304, September 1995.

[43] Richard B. Pember, John B. Bell, Phillip Colella, William Y. Crutchfield, and Michael L. Welcome. An adaptive Cartesian grid method for unsteady compressible flow in irregular regions. *J. Comput. Phys.*, 120(2):278 – 304, 1995.

[44] Charles S Peskin. Flow patterns around heart valves: A numerical method. *J. Comput. Phys.*, 10(2):252 – 271, 1972.

[45] Stephane Popinet. Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. *J. Comput. Phys.*, 190(2):572 – 600, 2003.

[46] J. J. Quirk. An alternative to unstructured grids for computing gas dynamics flows around arbitrarily complex two-dimensional bodies. *Computers and Fluids*, 23:125–142, 1994.

[47] P. O. Schwartz, J. Percelay, T. Ligocki, H. Johansen, D. T. Graves, P. Devendran, P. Colella, and E. Ateljevich. High accuracy embedded boundary grid generation using the divergence theorem. *Comm. App. Math. Comp. Sci.*, 2014. in press.

[48] D. Trebotich. Simulation of biological flow and transport in complex geometries using embedded boundary / volume-of-fluid methods. *Journal of Physics: Conference Series*, 78:012076, 2007.

[49] D. Trebotich, M. F. Adams, C. I. Steefel, S. Molins, and C. Shen. High resolution simulation of pore scale reactive transport processes associated with carbon sequestration. *Computing in Science and Engineering*, 2014. to appear Nov/Dec Leadership Computing Issue, DOI: 10.1109/MCSE.2014.77.

[50] D. Trebotich and P. Colella. A projection method for incompressible viscous flow on moving quadrilateral grids. *J. Comput. Phys.*, 166:191–217, 2001.

[51] D. Trebotich, P. Colella, G. H. Miller, A. Nonaka, T. Marshall, S. Gulati, and D. Liepmann. A numerical algorithm for complex biological flow in irregular microdevice geometries. In *Technical Proceedings of the 2004 Nanotechnology Conference and Trade Show*, volume 2, pages 470–473, 2004.

[52] D. Trebotich, G. H. Miller, and M. D. Bybee. A penalty method to model particle interactions in DNA-laden flows. *J. Nanosci. Nanotechnol.*, 8(7):3749–3756, 2008.

[53] D. Trebotich, G. H. Miller, P. Colella, D. T. Graves, D.F. Martin, and P. O. Schwartz. A tightly coupled particle-fluid model for DNA-laden flows in complex microscale geometries. In K. J. Bathe, editor, *Computational Fluid and Solid Mechanics*, pages 1018–1022. Elsevier, 2005.

[54] D. Trebotich, B. Van Straalen, D. T. Graves, and P. Colella. Performance of embedded boundary methods for CFD with complex geometry. *Journal of Physics: Conference Series*, 125:012083, 2008.

[55] E. H. Twizell, A. B. Gumel, and M. A. Arigu. Second-order, L_0-stable methods for the heat equation with time-dependent boundary conditions. *Advances in Computational Mathematics*, 6:333–352, 1996.

[56] B. van Leer. Towards the ultimate conservative difference scheme: A second-order sequel to Godunovs method. *J. Comput. Phys.*, 32:101–136, 1979.

[57] C. H. K. Williamson. Defining a universal and continuous Strouhal-Reynolds number relationship for the laminar vortex shedding of a circular cylinder. *Phys. Fluids*, 31:2742, 1988.

[58] C. H. K. Williamson. The existence of two stages in the transition to three-dimensionality of a cylinder wake. *Phys. Fluids*, 31:3165, 1988.

[59] C. H. K. Williamson. Vortex dynamics in the cylinder wake. *Ann. Rev. Fluid. Mech.*, 28:477–539, 1996.

[60] T. Ye, R. Mittal, H.S. Udaykumar, and W. Shyy. An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries. *J. Comput. Phys.*, 156(2):209 – 240, 1999.

[61] H.-Q. Zhang, U. Fey, B. R. Noack, M. König, and H. Eckelmann. On the transition of the cylinder wake. *Phys. Fluids*, 7:779, 1995.

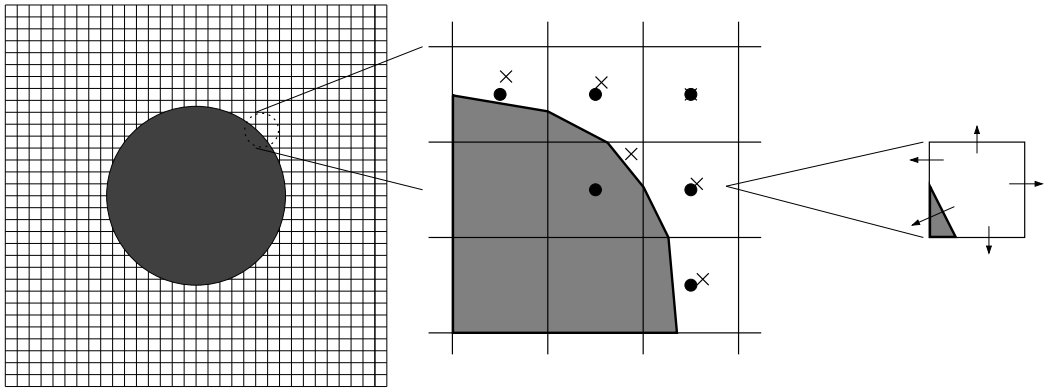


Fig. 1. Example of an irregular geometry on a Cartesian grid (left). Close-up view of embedded boundaries "cutting" regular cells (middle). Single cut cell showing boundary fluxes (right). Shaded area represents volume of cells excluded from domain. Dots represent cell–centers. X's represent centroids.

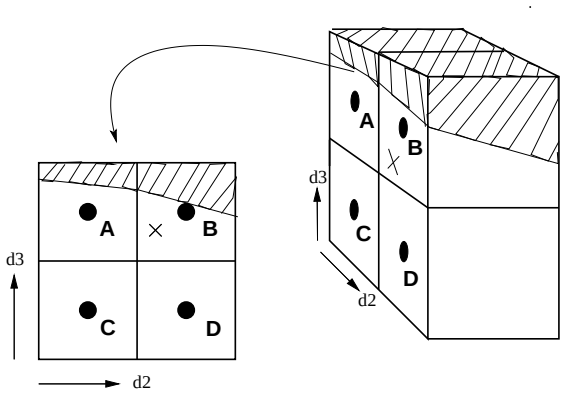


Fig. 2. 3D bilinear flux interpolation stencil showing the interpolation point marked by an "×" for a 3D face in $\hat{e}^1$ direction using face-centered points A, B, C and D.

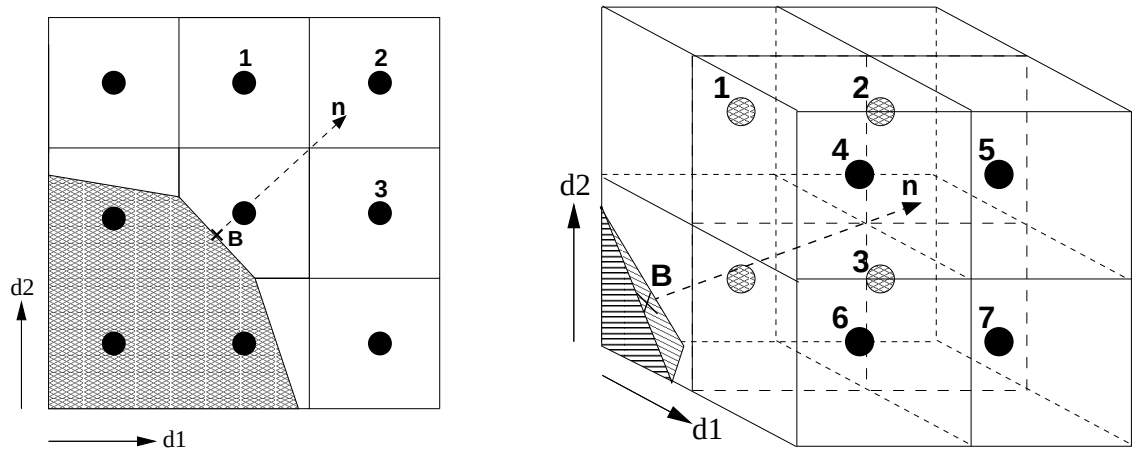


Fig. 3. Least squares stencil to obtain flux for Dirichlet boundary condition on embedded boundary in 2D (left) and 3D (right) with radius of 1.

Variable	$e_{8h\to4h}$	Order	$e_{4h\to2h}$	Order	$e_{2h\to h}$
u	1.781614e-05	1.981	4.513495e-06	1.990	1.135840e-06
v	1.788296e-05	1.981	4.529397e-06	1.991	1.139716e-06
$\nabla^x p$	1.245135e-04	1.998	3.117580e-05	1.999	7.798800e-06
$\nabla^y p$	1.245125e-04	1.998	3.117575e-05	1.999	7.798797e-06

Table 1
2D solution error convergence rates using L_1-norm for $h = \frac{1}{2048}$.

Variable	$e_{8h\to4h}$	Order	$e_{4h\to2h}$	Order	$e_{2h\to h}$
u	2.997859e-05	1.977	7.613610e-06	1.989	1.918581e-06
v	3.011462e-05	1.978	7.645604e-06	1.989	1.926329e-06
$\nabla^x p$	2.672975e-04	1.997	6.696069e-05	1.999	1.675127e-05
$\nabla^y p$	2.672958e-04	1.997	6.696059e-05	1.999	1.675127e-05

Table 2
2D solution error convergence rates using L_2-norm for $h = \frac{1}{2048}$.

Variable	$e_{8h\to4h}$	Order	$e_{4h\to2h}$	Order	$e_{2h\to h}$
u	1.234211e-04	1.980	3.129091e-05	1.990	7.878456e-06
v	1.237856e-04	1.980	3.138271e-05	1.990	7.899819e-06
$\nabla^x p$	1.396595e-03	1.995	3.504449e-04	1.998	8.771770e-05
$\nabla^y p$	1.396573e-03	1.995	3.504435e-04	1.998	8.771761e-05

Table 3
2D solution error convergence rates using L_∞-norm for $h = \frac{1}{2048}$.

Variable	$e_{8h\to4h}$	Order	$e_{4h\to2h}$	Order	$e_{2h\to h}$
u	1.071815e-03	1.865	2.941499e-04	1.948	7.623080e-05
v	1.069533e-03	1.864	2.937195e-04	1.948	7.613267e-05
w	1.068955e-03	1.865	2.933635e-04	1.948	7.603024e-05
$\nabla^x p$	1.593183e-02	1.806	4.556606e-03	1.933	1.192892e-03
$\nabla^y p$	1.592733e-02	1.805	4.556155e-03	1.933	1.192860e-03
$\nabla^z p$	1.593025e-02	1.806	4.556234e-03	1.933	1.192863e-03

Table 4
3D solution error convergence rates using L_1-norm for $h = \frac{1}{256}$.

Variable	$e_{8h\to4h}$	Order	$e_{4h\to2h}$	Order	$e_{2h\to h}$
u	2.294986e-03	1.846	6.382495e-04	1.900	1.709917e-04
v	2.288364e-03	1.847	6.361839e-04	1.900	1.704523e-04
w	2.287161e-03	1.847	6.357047e-04	1.900	1.703239e-04
$\nabla^x p$	4.846120e-02	1.748	1.442698e-02	1.922	3.808233e-03
$\nabla^y p$	4.845033e-02	1.748	1.442543e-02	1.921	3.808121e-03
$\nabla^z p$	4.843998e-02	1.748	1.442312e-02	1.921	3.807923e-03

Table 5
3D solution error convergence rates using L_2-norm for $h = \frac{1}{256}$.

Variable	$e_{8h\to4h}$	Order	$e_{4h\to2h}$	Order	$e_{2h\to h}$
u	1.891858e-02	1.841	5.280958e-03	1.781	1.536950e-03
v	1.903841e-02	1.849	5.285737e-03	1.803	1.514318e-03
w	1.911926e-02	1.846	5.319278e-03	1.793	1.534994e-03
$\nabla^x p$	4.893581e-01	1.565	1.653736e-01	1.868	4.529462e-02
$\nabla^y p$	4.880882e-01	1.562	1.652861e-01	1.868	4.528107e-02
$\nabla^z p$	4.886462e-01	1.564	1.653033e-01	1.868	4.528735e-02

Table 6
3D solution error convergence rates using L_∞-norm for $h = \frac{1}{256}$.

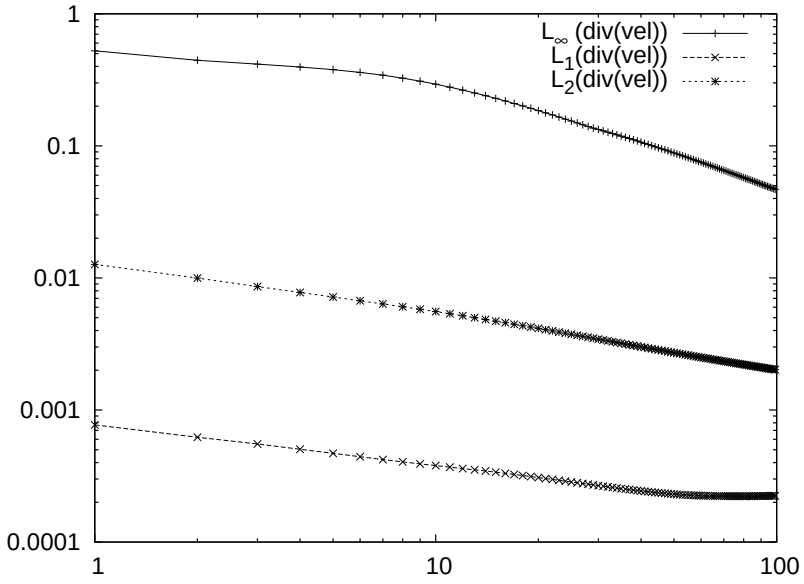


Fig. 4. Norms (L_1, L_2, L_∞) of $\kappa \mathbf{D}\vec{u}$ versus number of projection iterations in 2D test. $h = 1.0/256$

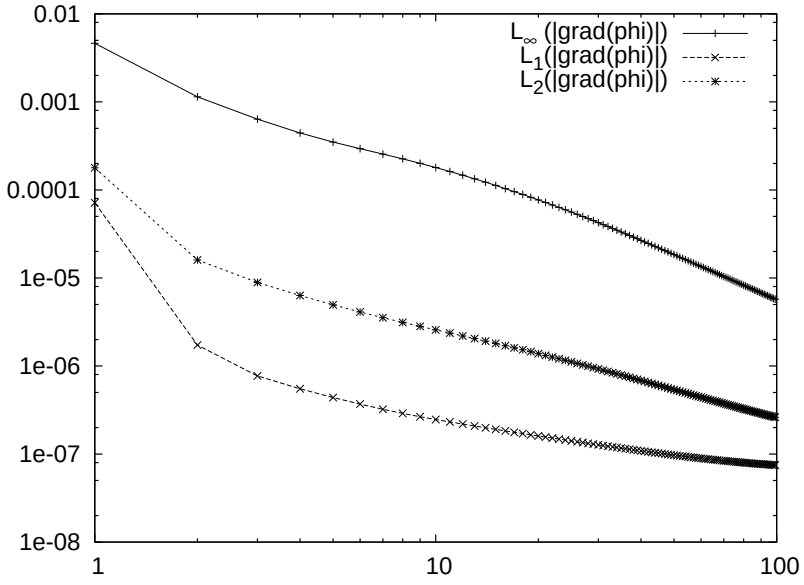


Fig. 5. Norms (L_1, L_2, L_∞) of $\nabla\phi$ versus number of projection iterations in 2D test. $h = 1.0/256$

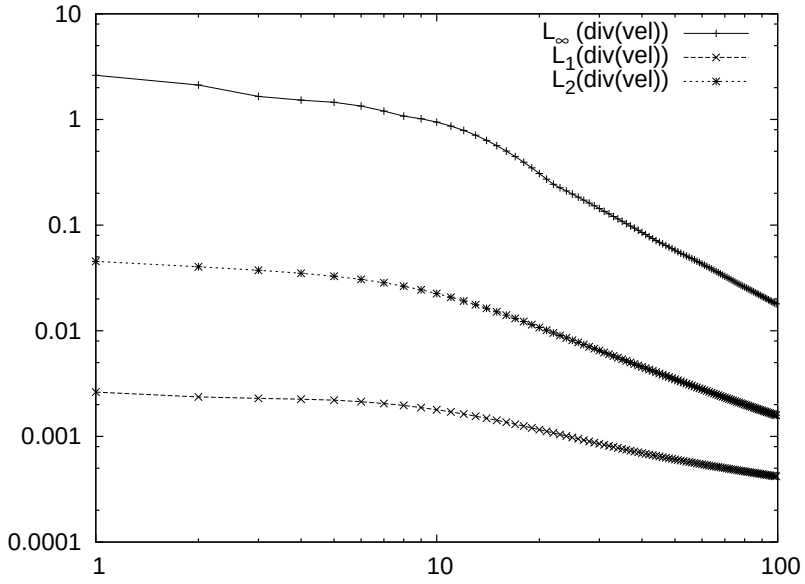


Fig. 6. Norms (L_1, L_2, L_∞) of $\kappa \mathbf{D}\vec{u}$ versus number of projection iterations in 3D test. $h = 1.0/64$

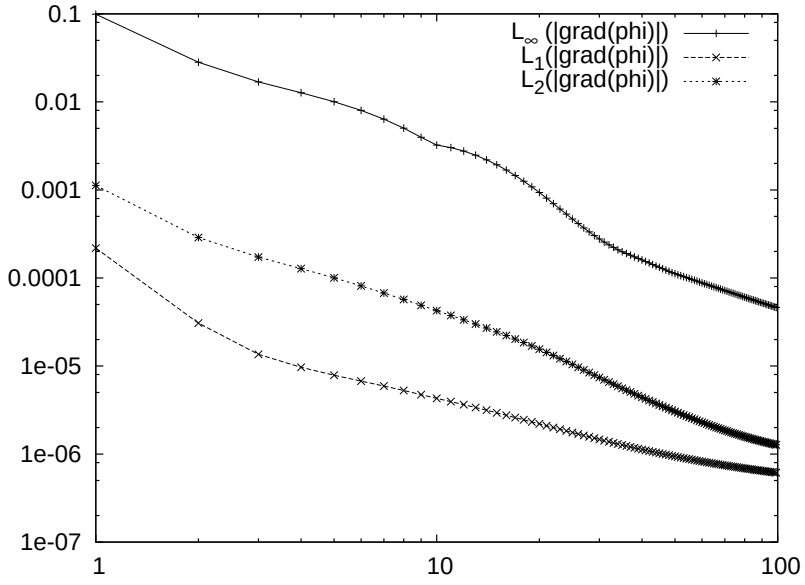
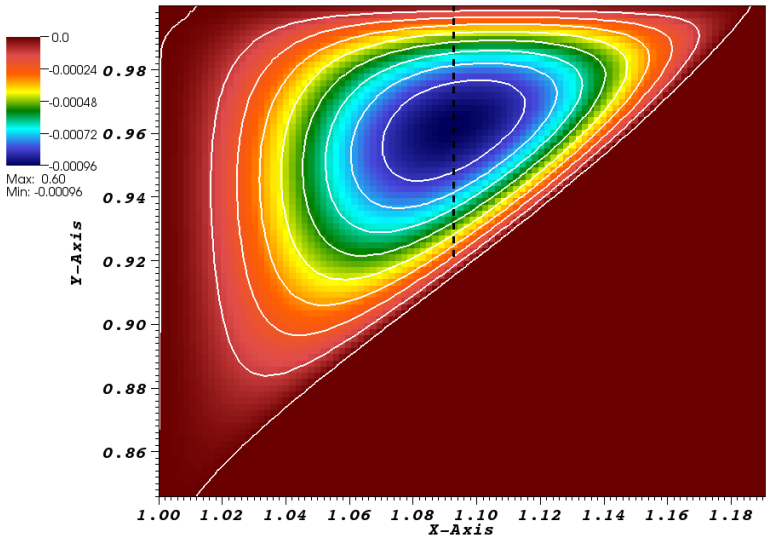
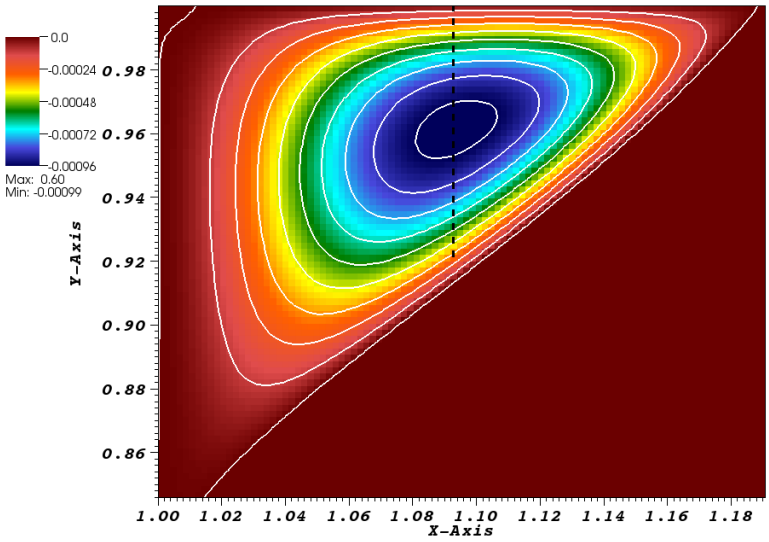


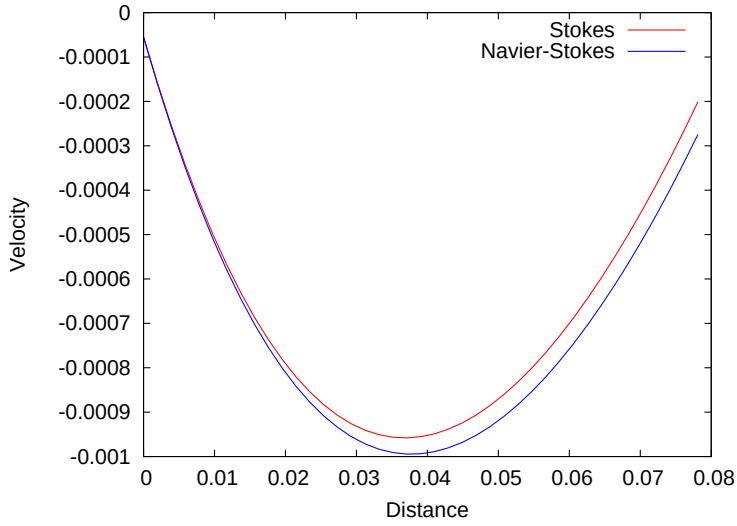
Fig. 7. Norms (L_1, L_2, L_∞) of $\nabla\phi$ versus number of projection iterations in 3D test. $h = 1.0/64$



(a)



(b)



(c)

Fig. 8. Zoomed in view of recirculation zone in the corner behind backward facing step for $Re = 0.1$. (a) Solution to Stokes equation. (b) Solution to Navier-Stokes equation. (c) Plots of velocity along dotted line in top figures.

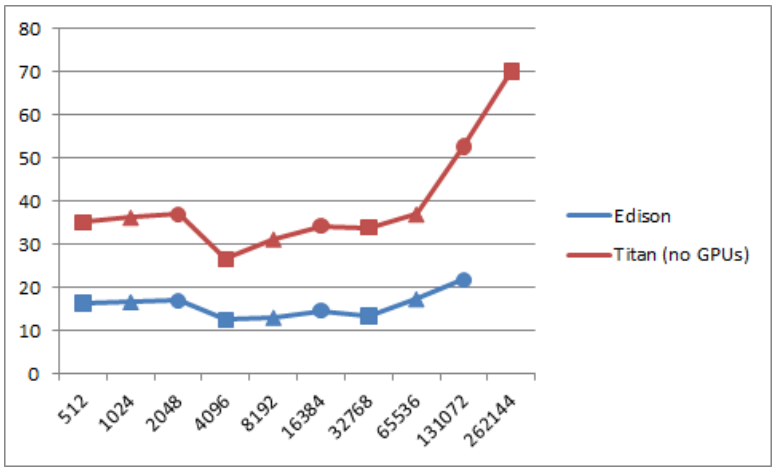
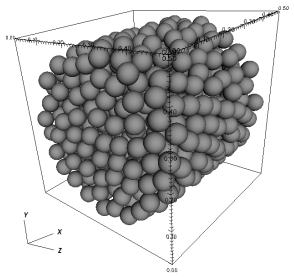
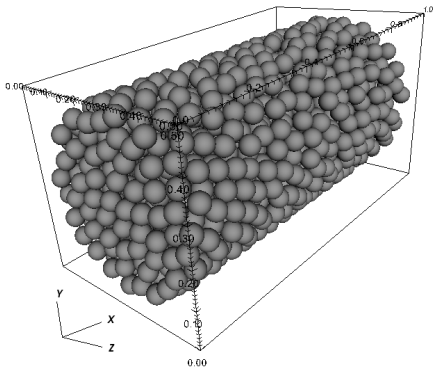


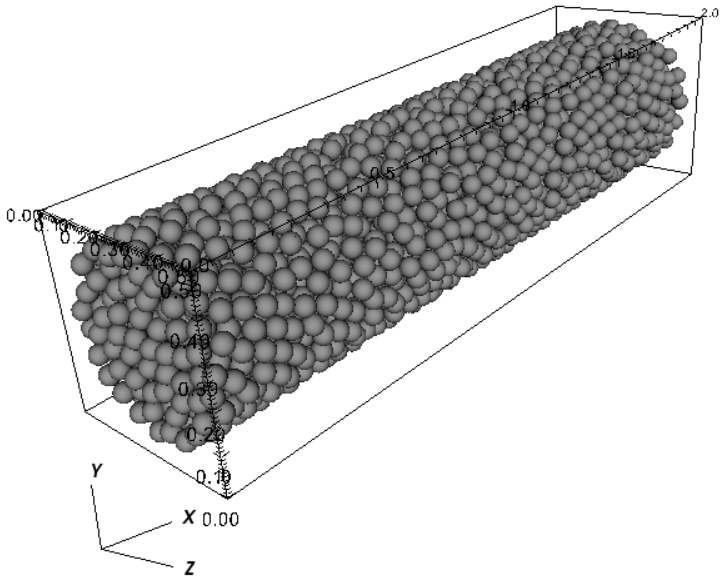
Fig. 9. Weak scaling on NERSC Cray XC30 (Edison) and OLCF Cray XK7 (Titan, no GPUs). Horizontal axis is concurrency, N, and vertical axis is average time in seconds per timestep.



(a)



(b)



(c)

Fig. 10. Packed cylinder geometries for weak scaling (replication) test in Table 9. (a) 1 cm long cylinder packed with 750 spheres, resolution $h \approx 2.44\mu$m ($2048 \times 2048 \times 2048$). This geometry is used for $N = 512, 4096, 32768, 261144$. (b) 2 cm long cylinder packed with 1500 spheres ($2048 \times 1024 \times 1024$), resolution $h \approx 4.88\mu$m. This geometry is used for $N = 1024, 8192, 65536$. (c) 4 cm long cylinder packed with 3000 spheres, resolution $h \approx 4.88\mu$m ($4096 \times 1024 \times 1024$). This geometry is used for $N = 2048, 16384, 131072$. Sphere radii = 250 μm.

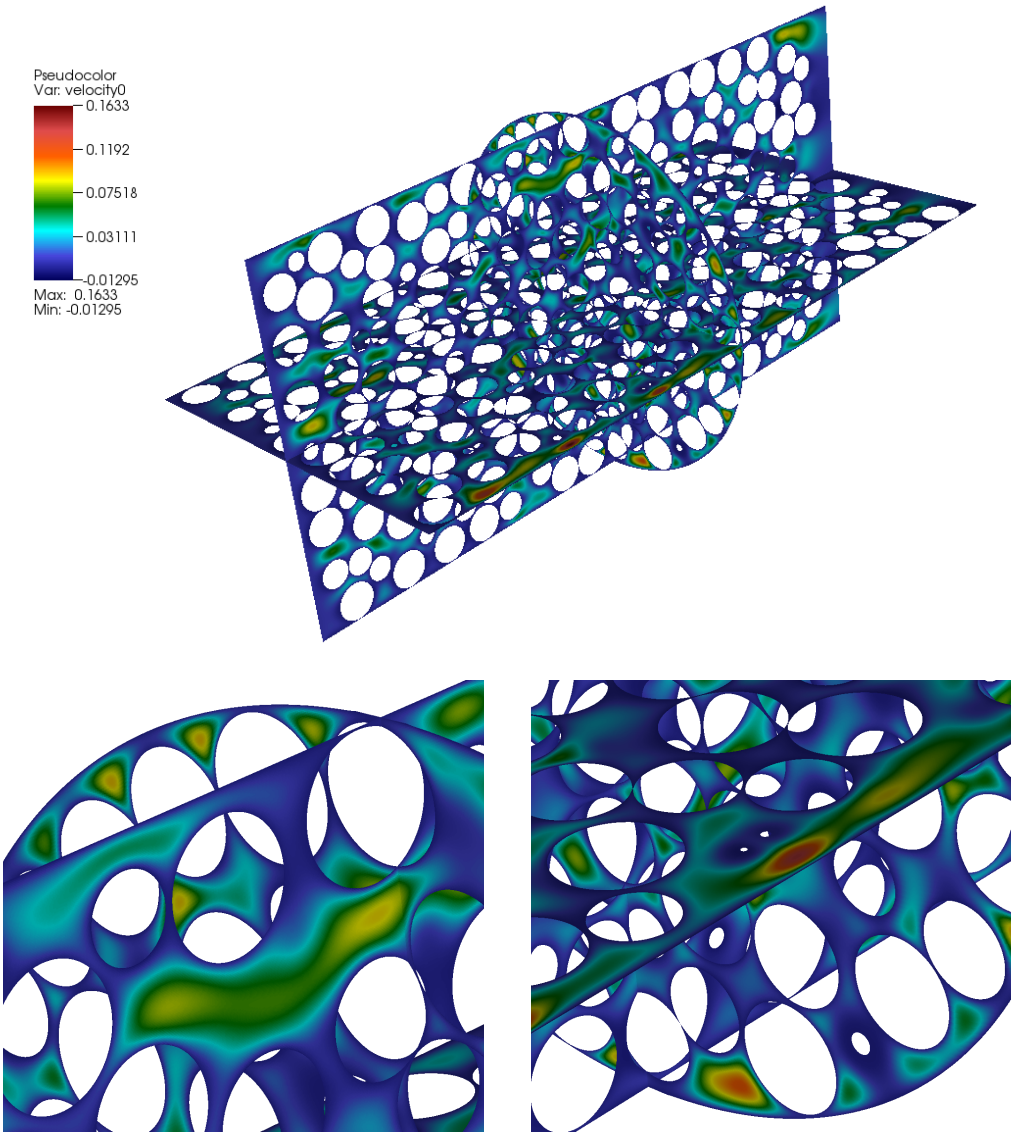


Fig. 11. Steady-state flow through the packed cylinder geometry in Figure 10b. Axial velocity shown with magnified views of top and bottom of cross-sectional slice. Inlet axial velocity is 0.01 cm/sec, $Re = 0.5$. Computation performed on NERSC Cray XC30 Edison using 65,536 processor cores.

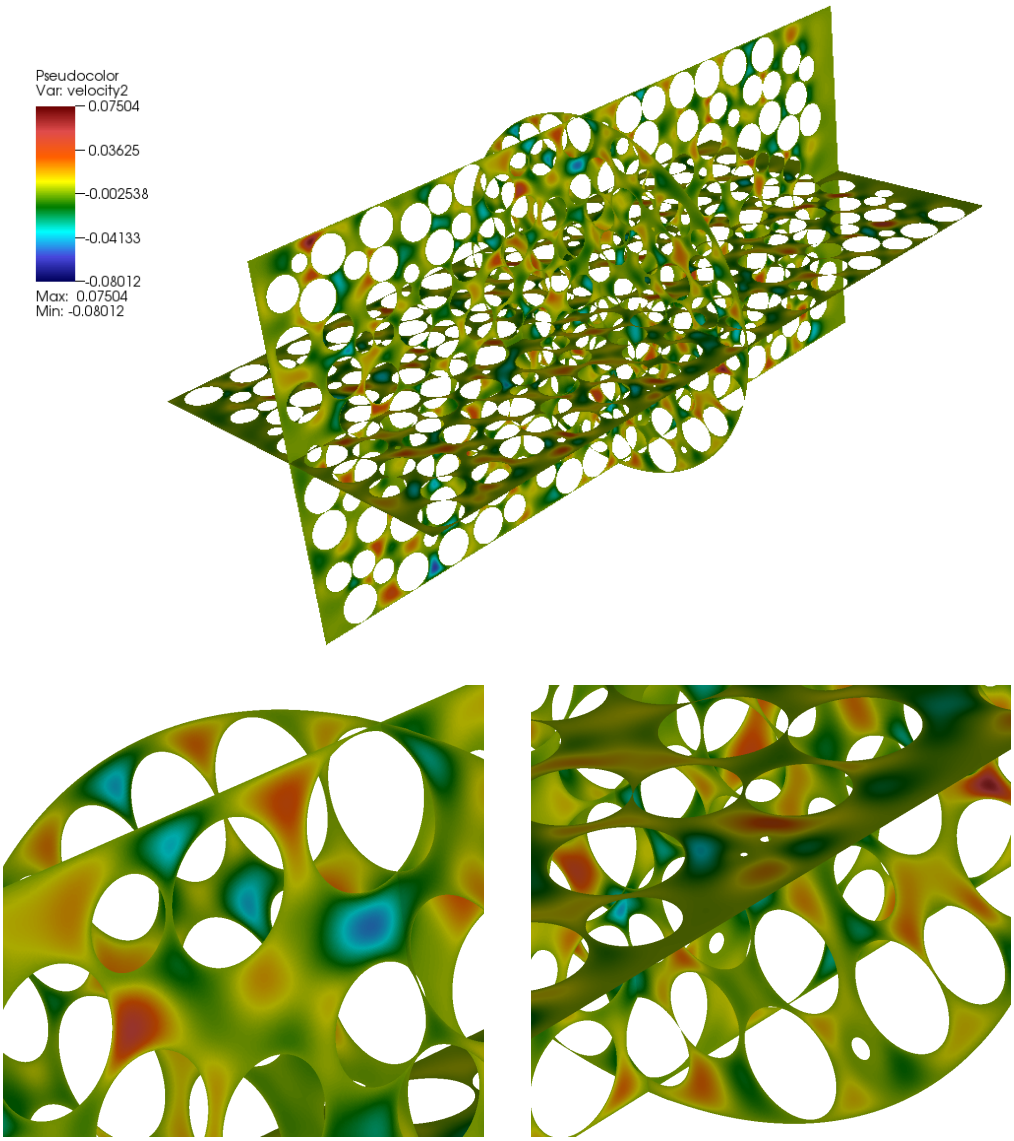


Fig. 12. Steady-state flow through the packed cylinder geometry in Figure 10b. Transverse (z) velocity shown with magnified views of top and bottom of cross-sectional slice. Inlet axial velocity is 0.01 cm/sec, $Re = 0.5$. Computation performed on NERSC Cray XC30 Edison using 65,536 processor cores.

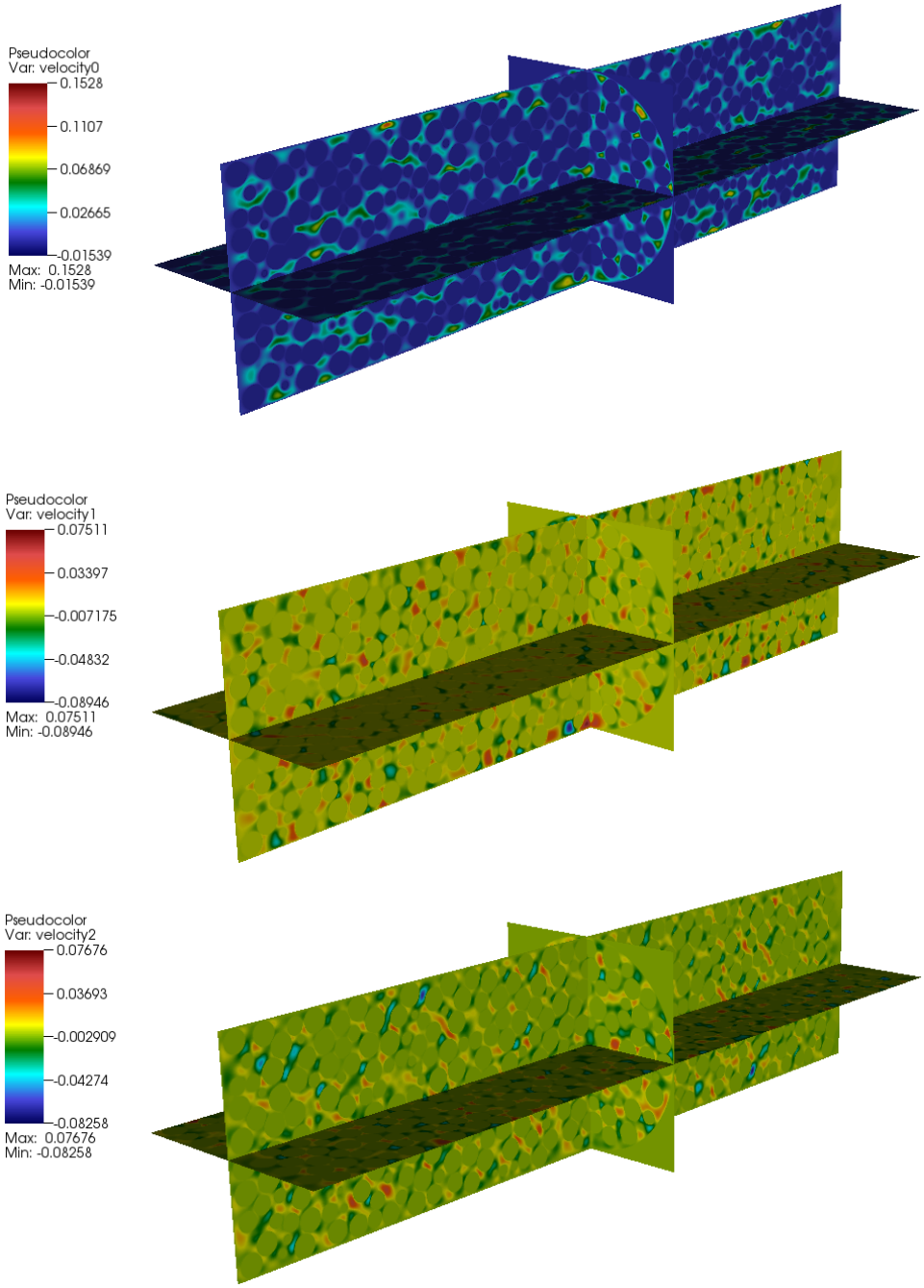


Fig. 13. Steady-state flow through the packed cylinder geometry in Figure 10c. From top to bottom, axial (x) and transverse (y) and (z) velocities. Inlet axial velocity is 0.01 cm/sec, $Re = 0.5$. Spheres have not been voided in the visualization, unlike in Figures 11 and 12. Computation performed on OLCF Titan using 131,072 processor cores.

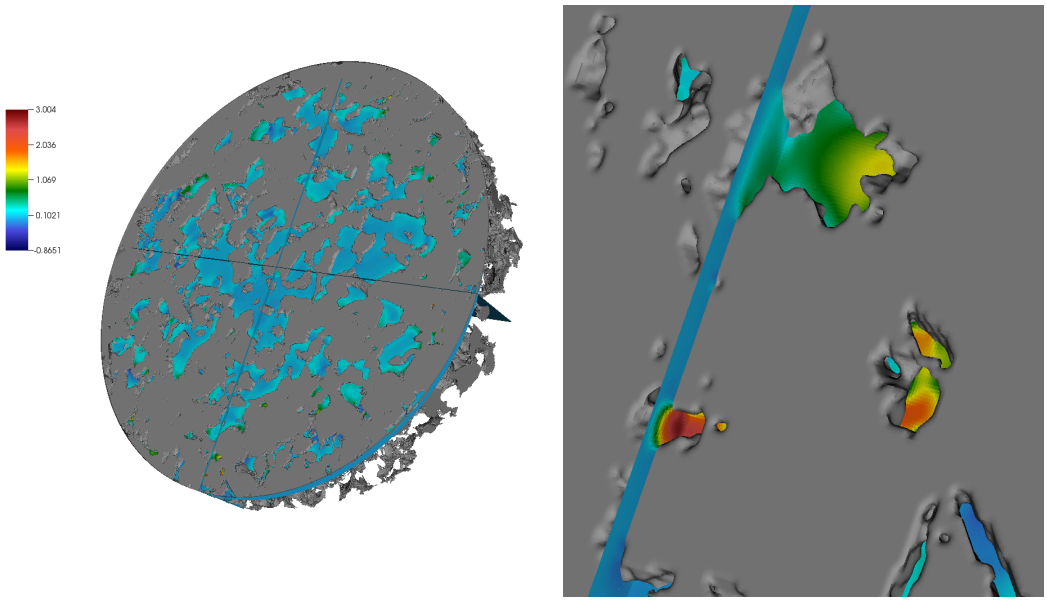


Fig. 14. Steady-state flow through Bedford limestone. Embedded boundary grid in gray with $z-$velocity data (left), and magnification of bottom of disk (right). $h = 43$ nm, $2048 \times 2048 \times 320$ total grid cells, 29% porosity. Inlet velocity is 0.0376 cm/sec. Computation performed on OLCF Titan using 40,960 processor cores.

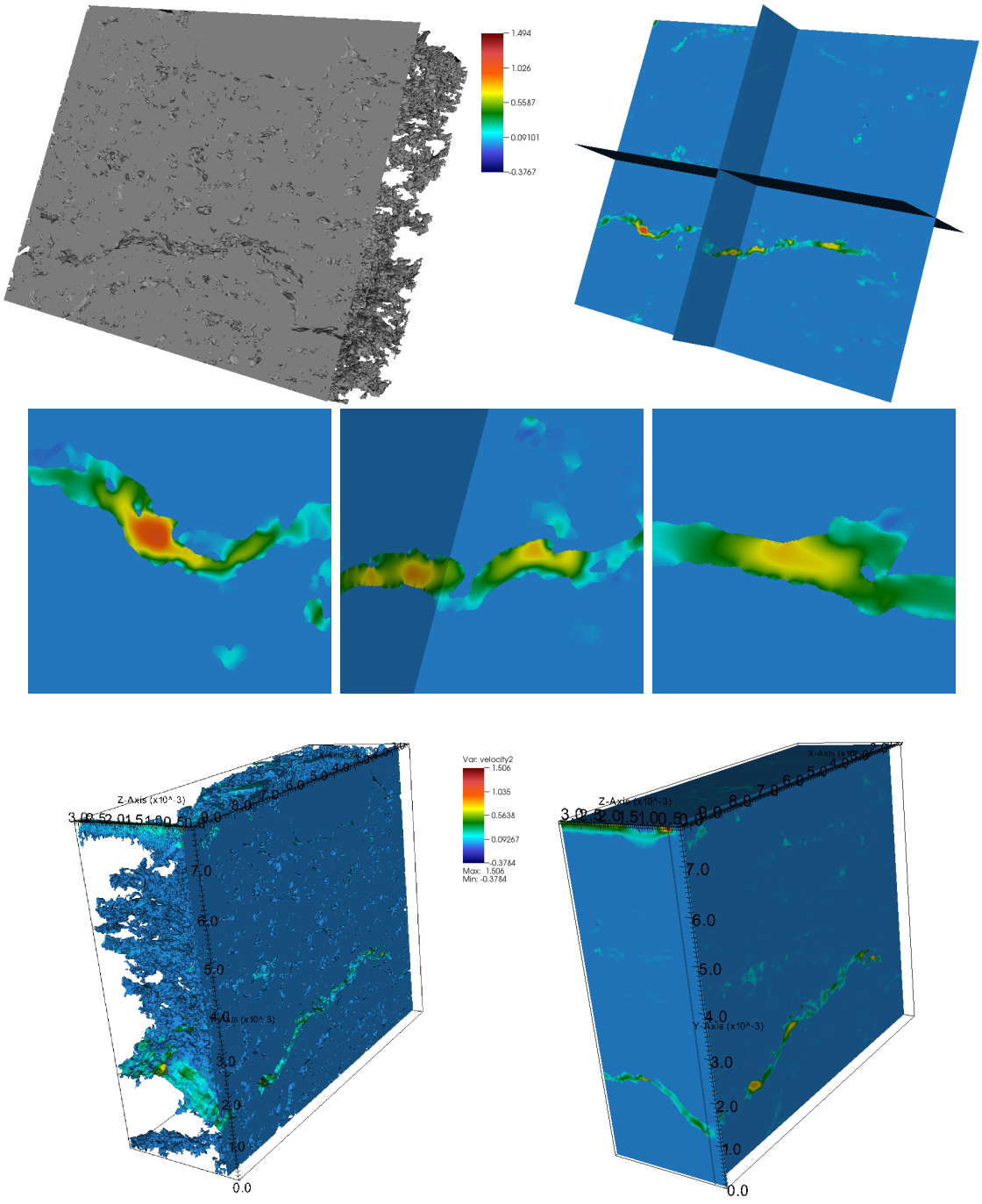
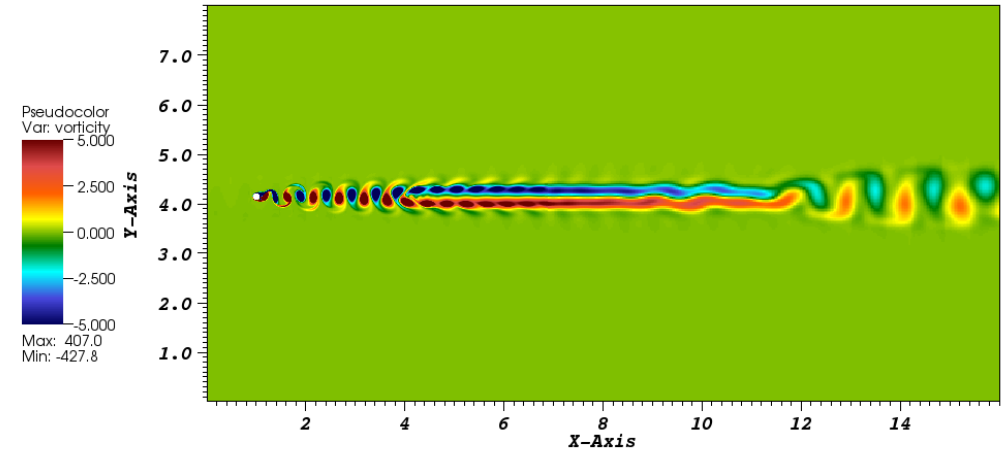
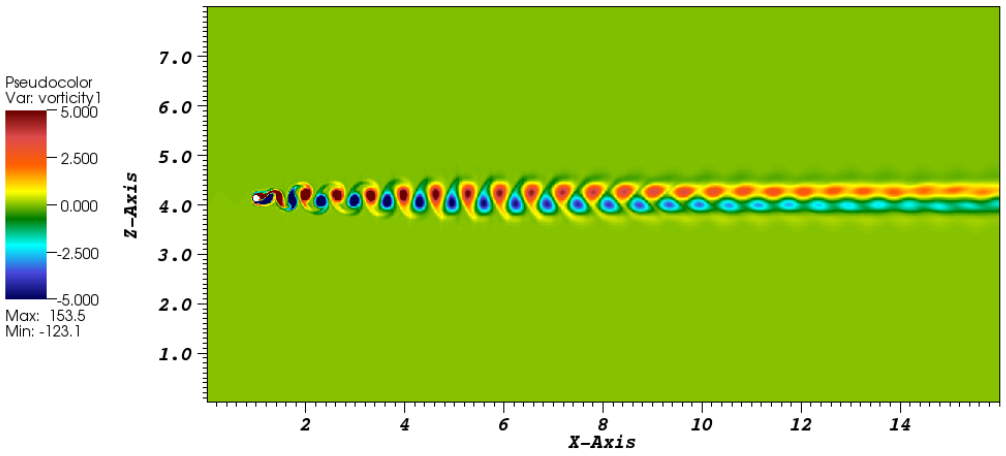


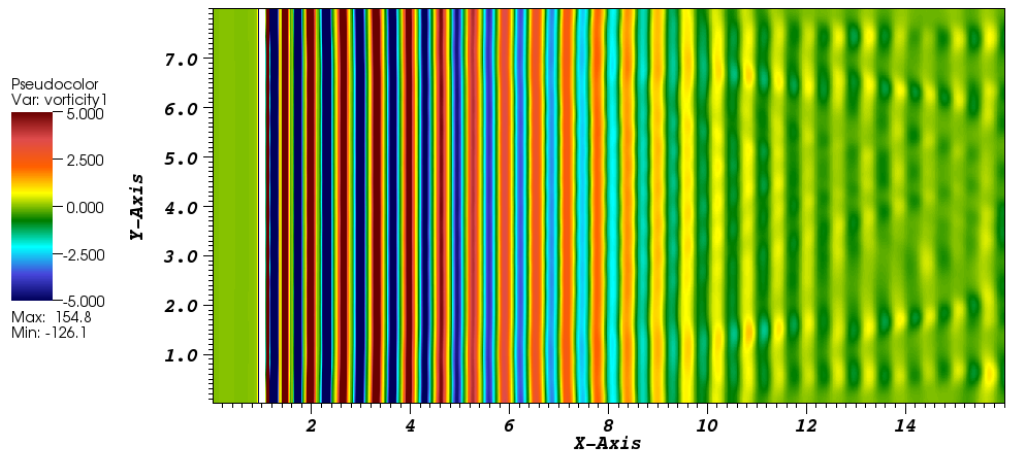
Fig. 15. Steady-state flow through fractured shale. Embedded boundary grid in gray (top, left) with slices at mid-planes of $z-$velocity (top, right), magnifications (middle) and rotated view of boundary data (bottom, left) with slice planes near maximum velocity location (bottom, right). $h = 48.4$ nm, $1920 \times 1600 \times 640$ total grid cells, 18% porosity. Inlet velocity is 0.008 cm/sec. Computation performed on NERSC Hopper using 60,000 processor cores.



(a)



(b)



(c)

Fig. 16. Flow past a cylinder at $Re = 300$. (a) 2D simulation of vorticity; (b) 3D simulation of y-vorticity in x-z plane at $z = 4.125$; (c) 3D simulation of y-vorticity in x-y plane at $y = 4$. Cylinder has radius of $r = 0.0625$ and is shown in white centered at $x = 1$, $y = 4.125$ in 2D, and $x = 1$, $y = 4$, $z = 4.125$ in 3D. Time simulated is 98 seconds.

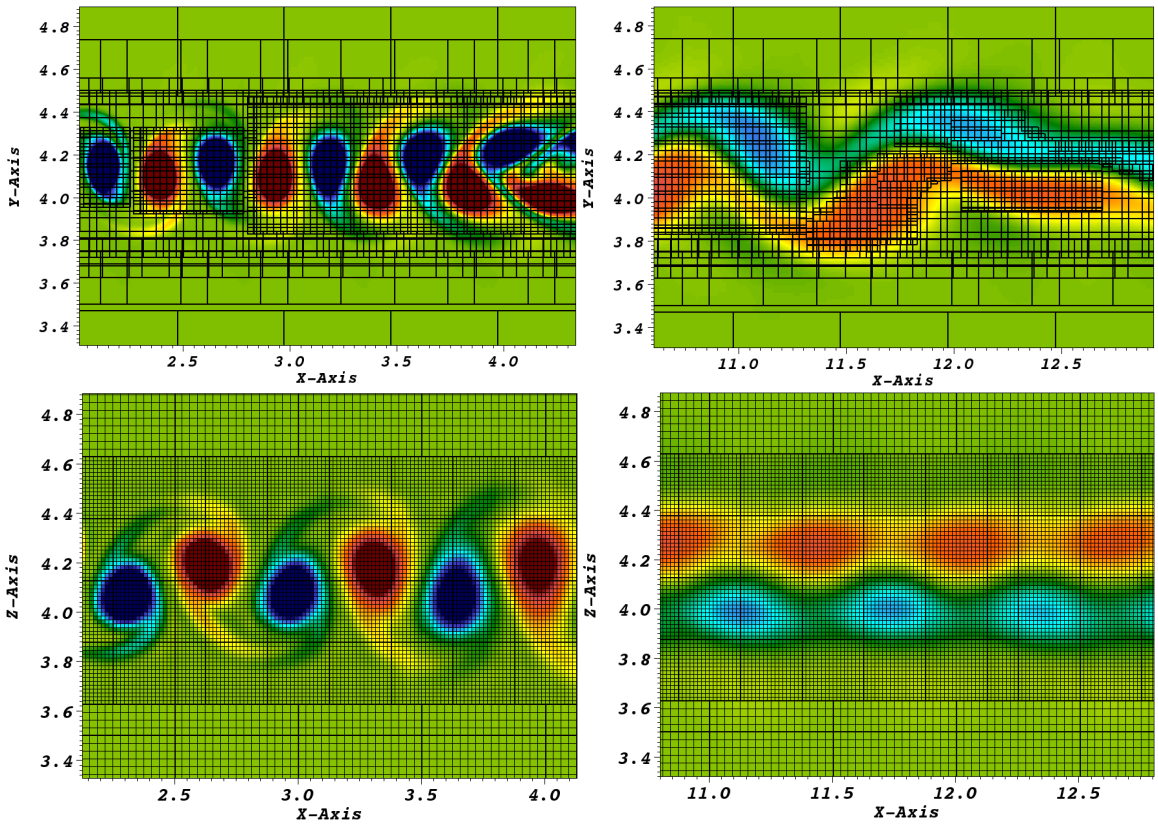


Fig. 17. Zoomed in view of Figure 16 showing AMR hierarchy. (top) 2D vorticity at left and right sections in the wake with grid block outlines for 4 levels of refinement, factor of 2. Each block has a maximum of 16^2 cells (cells not shown). Finest level ($h = 1/512$) is gridded on 5% of the total domain. (bottom) 3D y-vorticity in x-z plane at $y = 4$ at left and right sections in the wake with grid block outlines and cells for 2 levels of refinement, factor of 2. Each block contains 16^3 cells in 3D (cells shown for x-z plane). Finest level ($h = 1/64$) is gridded on 12% of the total domain.

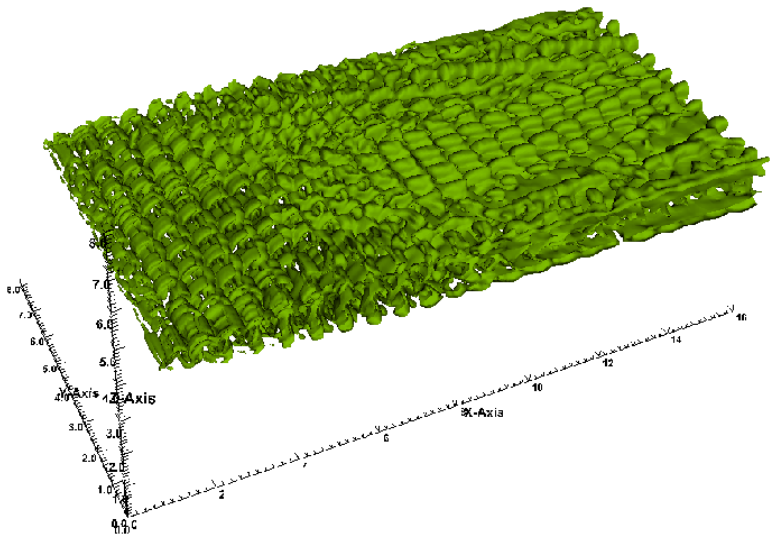
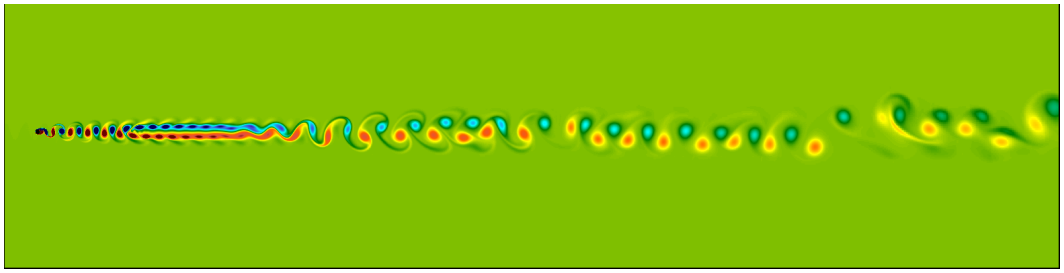
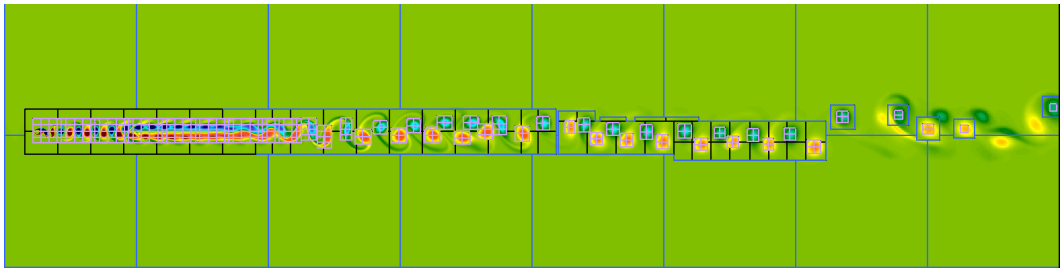


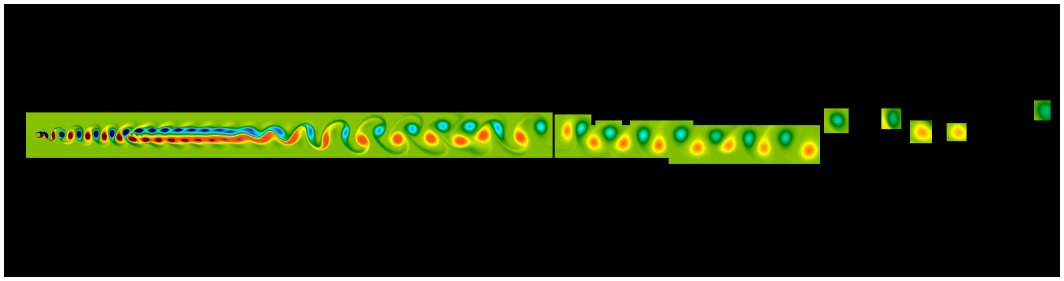
Fig. 18. 3D isocontour ($\omega_z = 0.001$) of z-vorticity for flow past a cylinder at $Re = 300$ and $t = 98$ seconds. Isocontour is contained by the finest level of grid blocks.



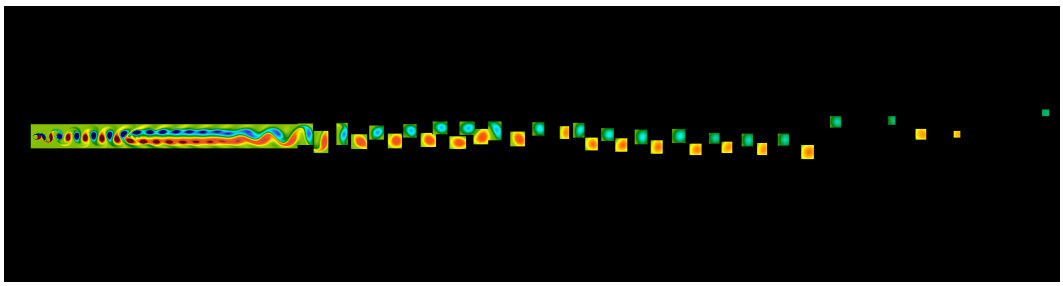
(a)



(b)



(c)



(d)

Fig. 19. 2D flow past a cylinder at $Re = 300$ with extended wake ($l = 32$). Vorticity in (a), with AMR hierarchy of boxes enclosing the wake in (b), and two finest levels in (c) and (d). Cylinder has radius of $r = 0.0625$ and is shown as a black spot near inlet. Wake extends approximately 250 cylinder diameters. Time simulated is 112 seconds.

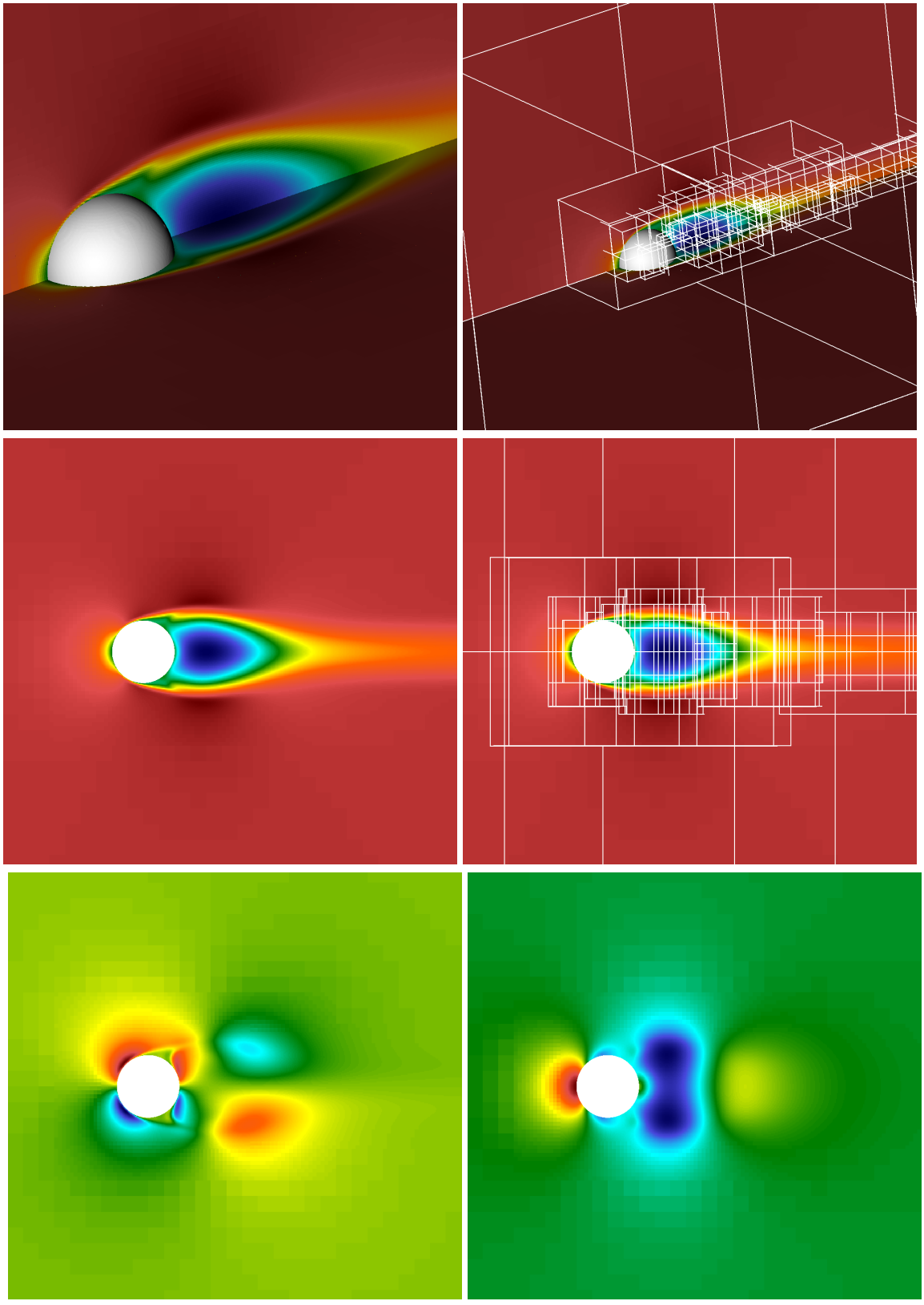


Fig. 20. Flow past a sphere, $Re = 600$. (top) Velocity in x-direction shown in 3D, with AMR grids (right). (middle) Velocity in x-direction in x-y plane, with grids (right). (bottom) Velocity in y-direction (left) and pressure in x-y plane (right). Base grid is $512 \times 256 \times 256$ with 2 additional levels of refinement, factor of 4. Domain is 16 cm long and 8 cm wide by 8 cm wide. Sphere diameter is 0.125 cm. Inlet velocity is 2 cm/sec. Simulated time is 2 seconds and early in wake formation with no vortex shedding. Simulation performed on 8192 nodes, 1 rank per node on ALCF BGQ Mira.

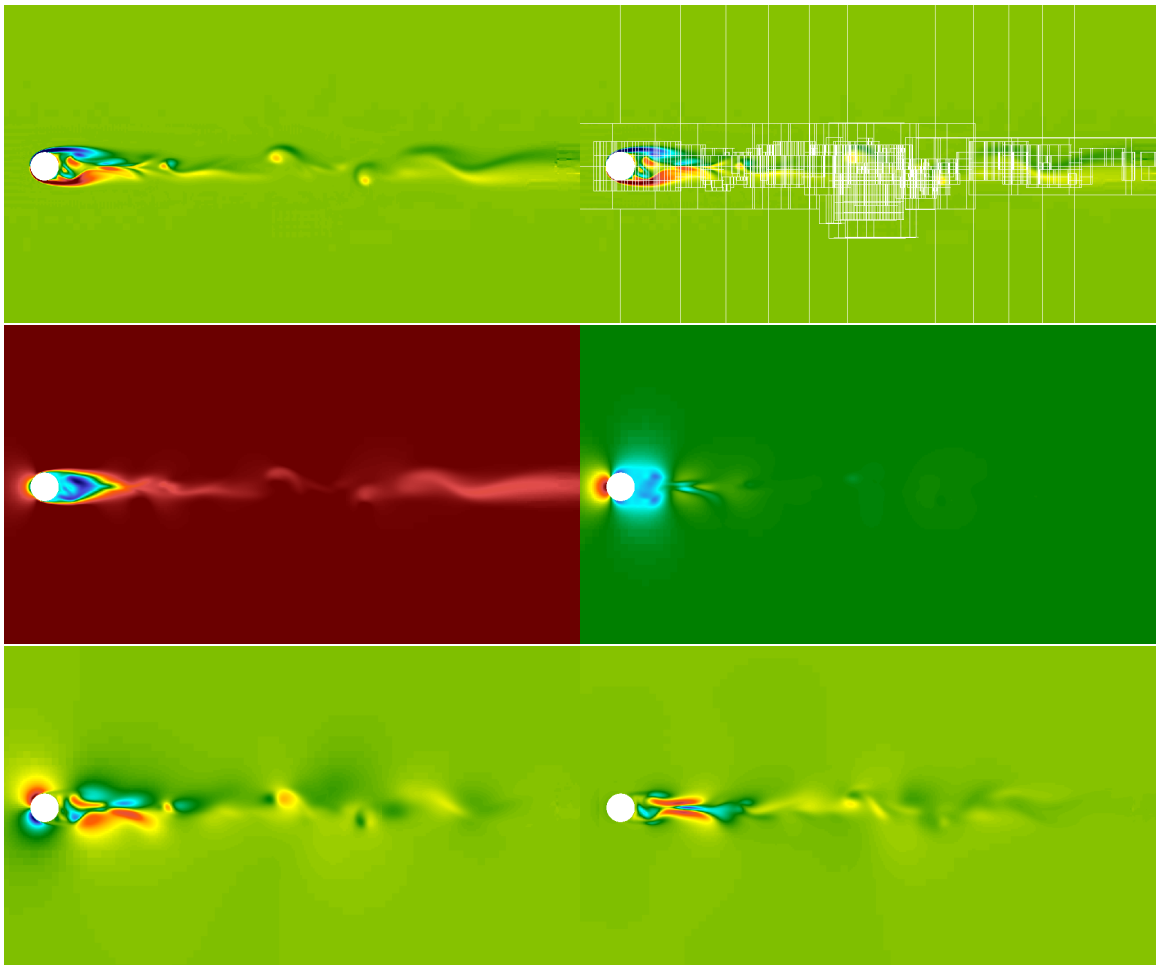


Fig. 21. Flow past a sphere, $Re = 600$. (top) Vorticity in z-direction (left), with AMR grids (right). (middle) Velocity in x-direction (left), pressure (right). (bottom) Velocity in y-direction (left) and velocity in z-direction (right). Base grid is $512 \times 256 \times 256$ with 2 additional levels of refinement, factor of 4. Domain is 16 cm long and 8 cm by 8 cm in cross section. Sphere diameter is 0.125 cm. Inlet velocity is 2 cm/sec. Simulated time is 2.5 seconds and early in wake formation but vortices have begun to shed. All plots shown in x-y plane. Simulation performed on 8192 nodes, 1 rank per node on ALCF BGQ Mira.

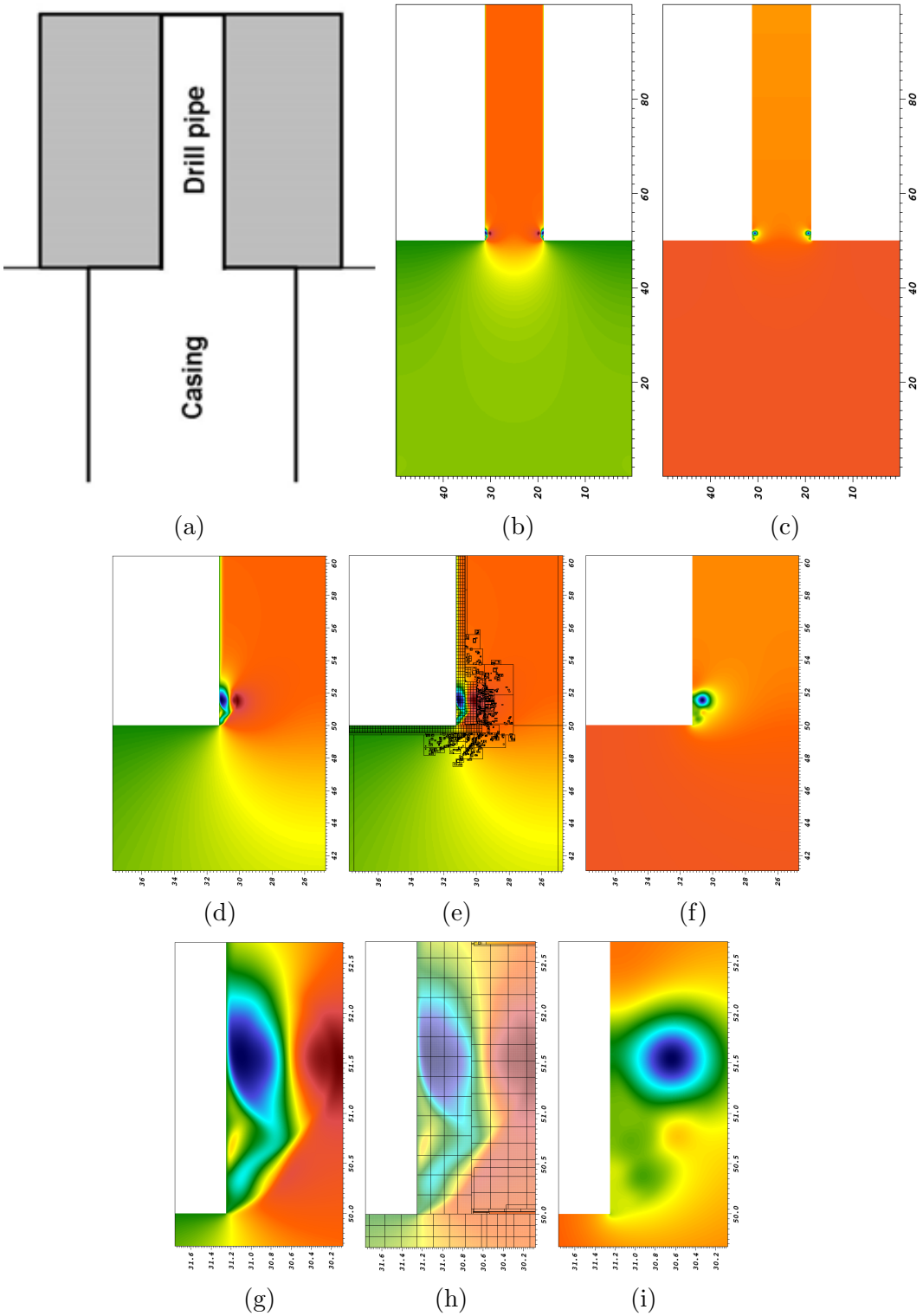


Fig. 22. Turbulent flow in 2D contraction, $Re \approx 6300$. (a) Conceptual model for oil flow entering a hypothetical failed blowout preventer depicted as a wellbore contracted into a (stuck) drill pipe. (b) Axial fluid velocity in 100 cm long by 50 cm wide section of contraction. (c) Pressure. (d) Increased magnification of velocity near contraction corner, with finer levels of AMR boxes (e). (f) Increased magnification of pressure. (g) Increased magnification of velocity, with finer levels of AMR boxes and mesh (h). (i) Increased magnification of pressure. Velocity range is -65.52 cm/sec (blue) to 84.66 cm/sec (red). Inlet average velocity is 10 cm/sec. Pressure range is -6662 bar (blue) to 3398 bar (red). Base grid is 2048×1024 with 3 additional levels of refinement by factor of 4. $t = 0.125$ secs. Computations performed on 8192 cores at NERSC.